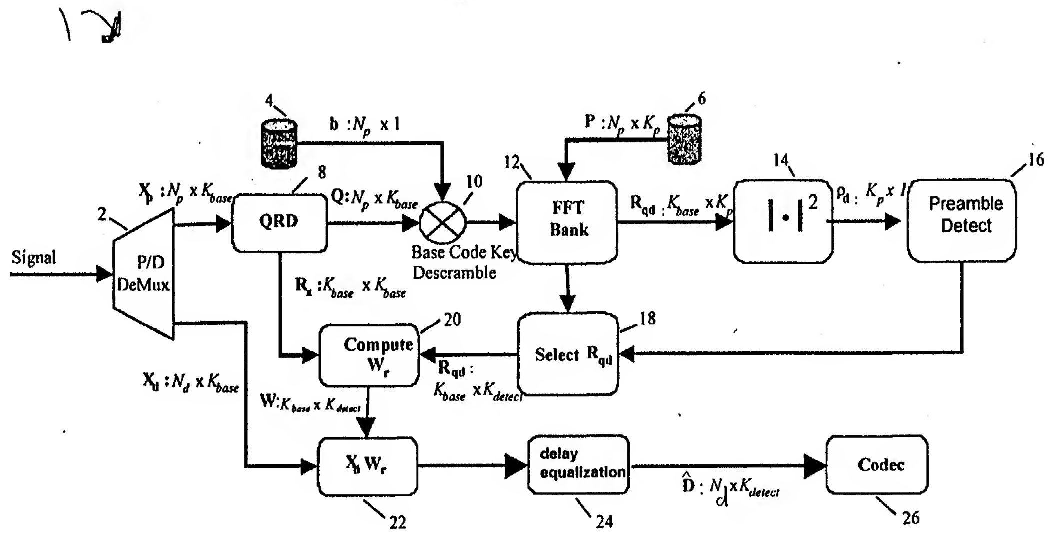
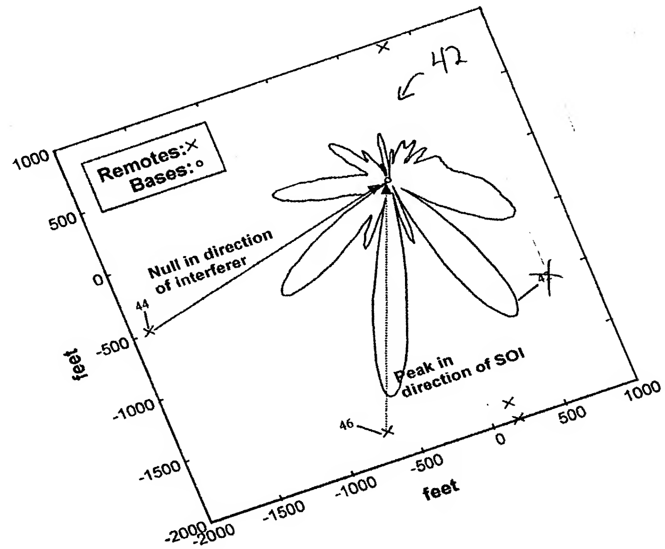

1
Signal
2
P/D DeMux
$X_p : N_p \times K_{base}$
$X_d : N_d \times K_{base}$
4
$b : N_p \times 1$
8
QRD
$Q : N_p \times K_{base}$
10
Base Code Key Descramble
$R_x : K_{base} \times K_{base}$
12
FFT Bank
6
$P : N_p \times K_p$
$R_{qd} : K_{base} \times K_p$
14
$|\cdot|^2$
$\rho_d : K_p \times 1$
16
Preamble Detect
18
Select R_{qd}
20
Compute W_r
$R_{qd} : K_{base} \times K_{detect}$
$W : K_{base} \times K_{detect}$
$X_d W_r$
22
delay equalization
24
$\hat{D} : N_d \times K_{detect}$
Codec
26

FIG. 1



Remotes: ×
Bases: ∘
Null in direction of interferer
Peak in direction of SOI
feet
feet
42
44
46
1000
500
0
-500
-1000
-1500
-2000

FIG. ~~4~~ 2

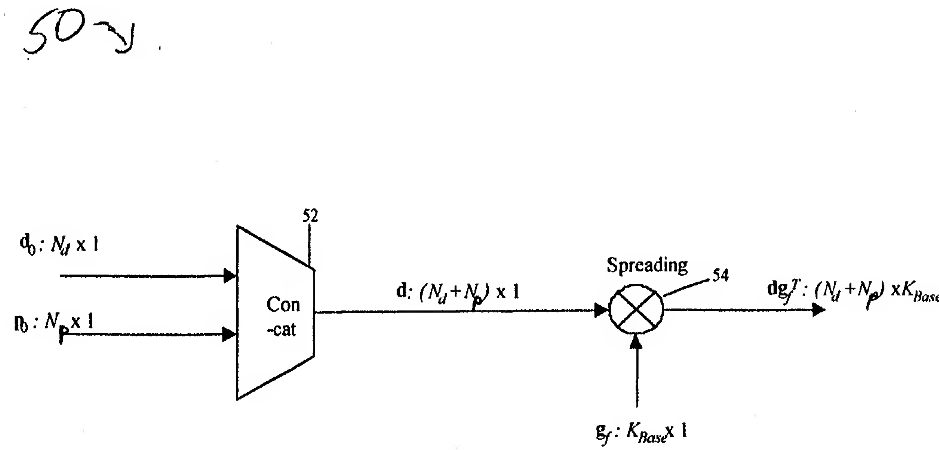
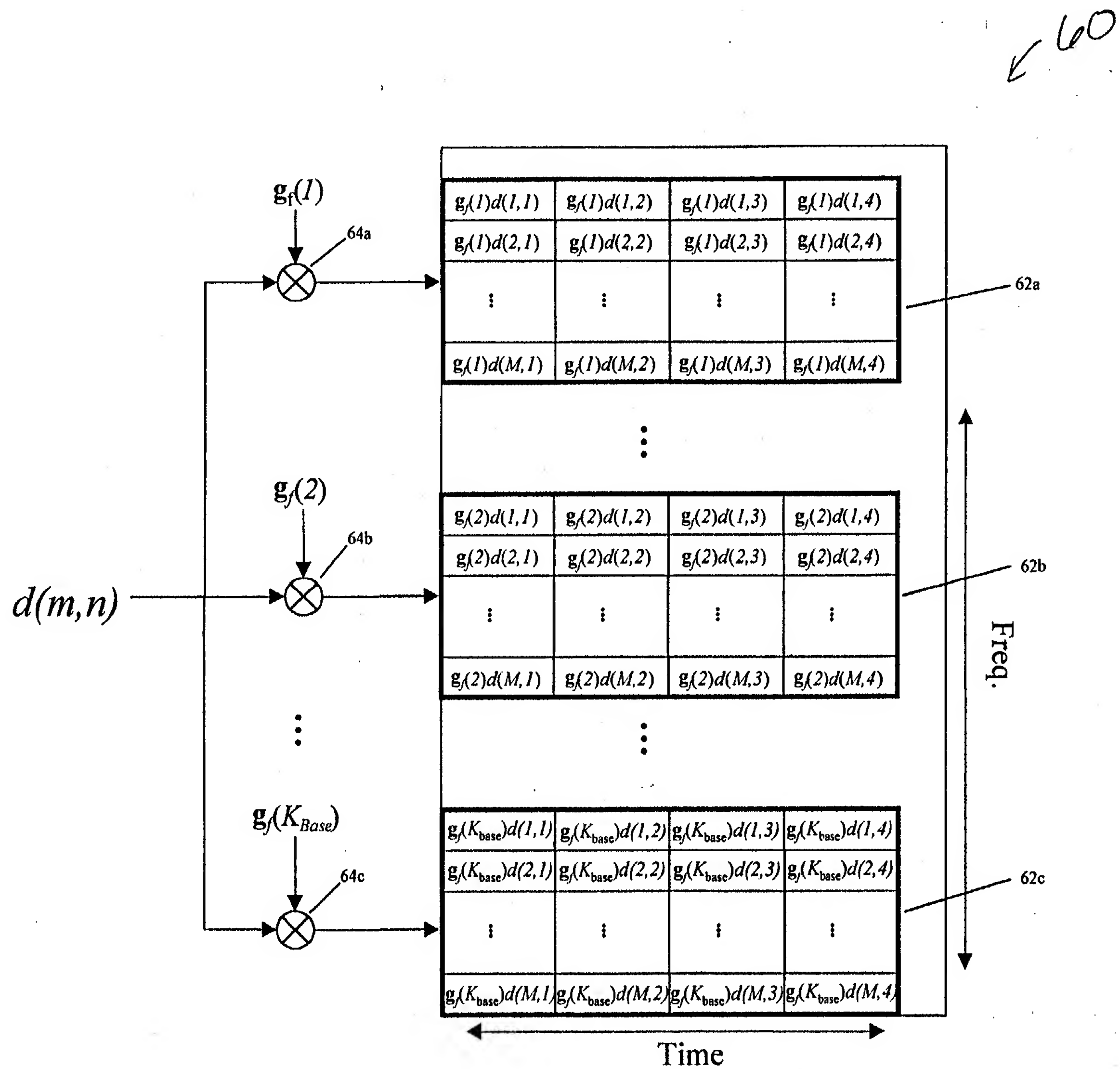

50
$\mathbf{d}_0 : N_d \times 1$
$\mathbf{n}_0 : N_p \times 1$
Con -cat
52
$\mathbf{d} : (N_d+N_p) \times 1$
Spreading
54
$\mathbf{g}_f : K_{Base} \times 1$
$\mathbf{d}\mathbf{g}_f^T : (N_d+N_p) \times K_{Base}$

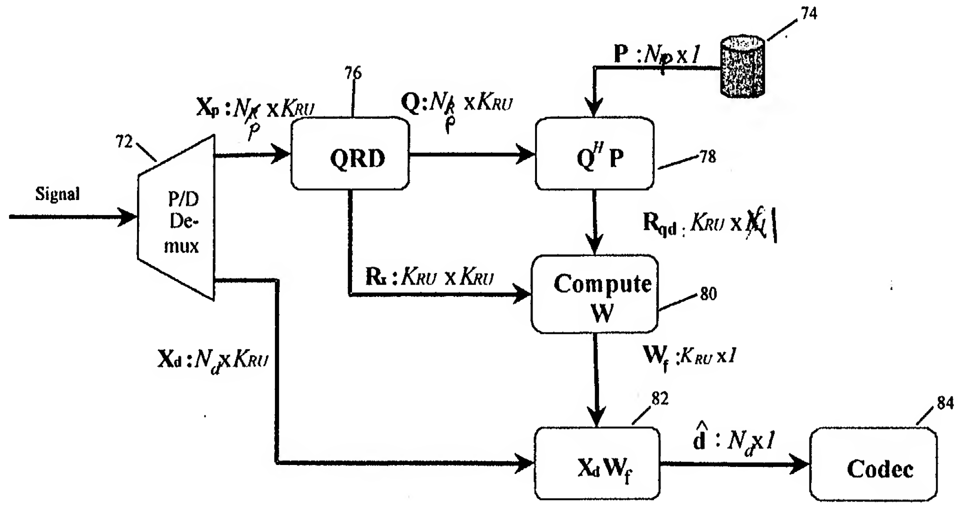
FIG. 5



60
d(m,n)
g_f(1)
64a
g_f(2)
64b
g_f(K_Base)
64c
62a
62b
62c
g_f(1)d(1,1) g_f(1)d(1,2) g_f(1)d(1,3) g_f(1)d(1,4)
g_f(1)d(2,1) g_f(1)d(2,2) g_f(1)d(2,3) g_f(1)d(2,4)
g_f(1)d(M,1) g_f(1)d(M,2) g_f(1)d(M,3) g_f(1)d(M,4)
g_f(2)d(1,1) g_f(2)d(1,2) g_f(2)d(1,3) g_f(2)d(1,4)
g_f(2)d(2,1) g_f(2)d(2,2) g_f(2)d(2,3) g_f(2)d(2,4)
g_f(2)d(M,1) g_f(2)d(M,2) g_f(2)d(M,3) g_f(2)d(M,4)
g_f(K_base)d(1,1) g_f(K_base)d(1,2) g_f(K_base)d(1,3) g_f(K_base)d(1,4)
g_f(K_base)d(2,1) g_f(K_base)d(2,2) g_f(K_base)d(2,3) g_f(K_base)d(2,4)
g_f(K_base)d(M,1) g_f(K_base)d(M,2) g_f(K_base)d(M,3) g_f(K_base)d(M,4)
Freq.
Time

FIG. ~~6~~ 4

70



Signal
72
P/D De-mux
$X_p : N_p \times K_{RU}$
76
QRD
$Q : N_p \times K_{RU}$
74
$P : N_p \times 1$
$Q^H P$
78
$R_x : K_{RU} \times K_{RU}$
$R_{qd} : K_{RU} \times 1$
Compute W
80
$W_f : K_{RU} \times 1$
$X_d : N_d \times K_{RU}$
82
$X_d W_f$
$\hat{d} : N_d \times 1$
84
Codec

FIG. ~~7~~ 5

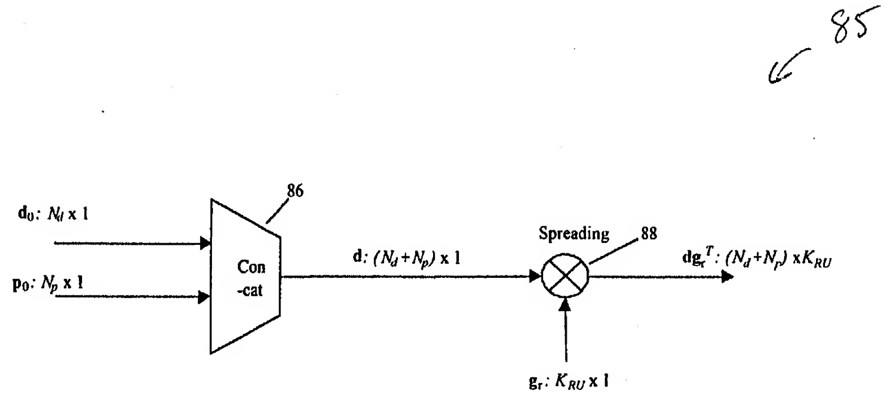
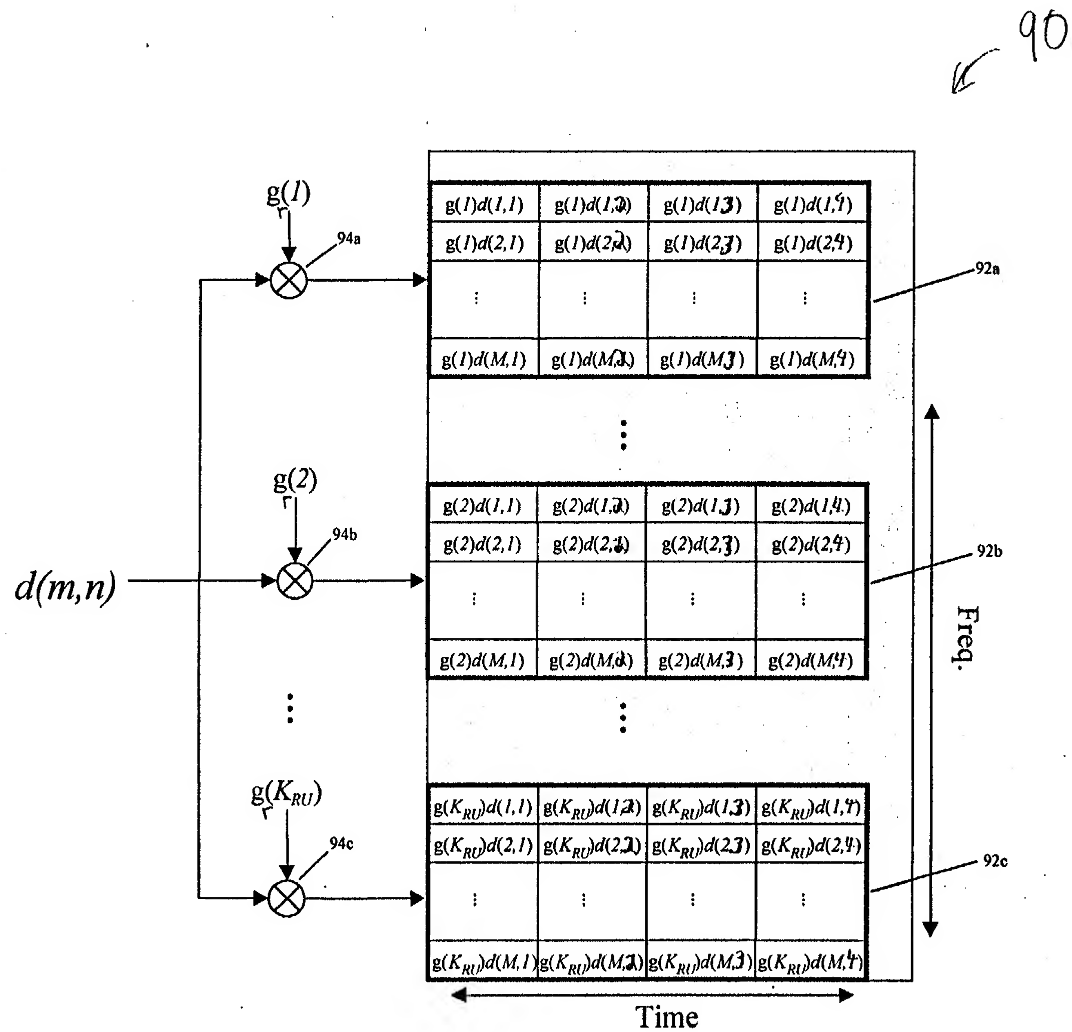

85
86
d_0: N_d x 1
p_0: N_p x 1
Con
-cat
d: (N_d+N_p) x 1
Spreading
88
dg_r^T: (N_d+N_p) xK_{RU}
g_r: K_{RU} x 1

FIG. ~~8~~ 6



90
d(m,n)
g(1)
94a
g(2)
94b
g(K_RU)
94c
92a
92b
92c
g(1)d(1,1)
g(1)d(2,1)
g(1)d(M,1)
g(2)d(1,1)
g(2)d(2,1)
g(2)d(M,1)
Freq.
Time

FIG. 9

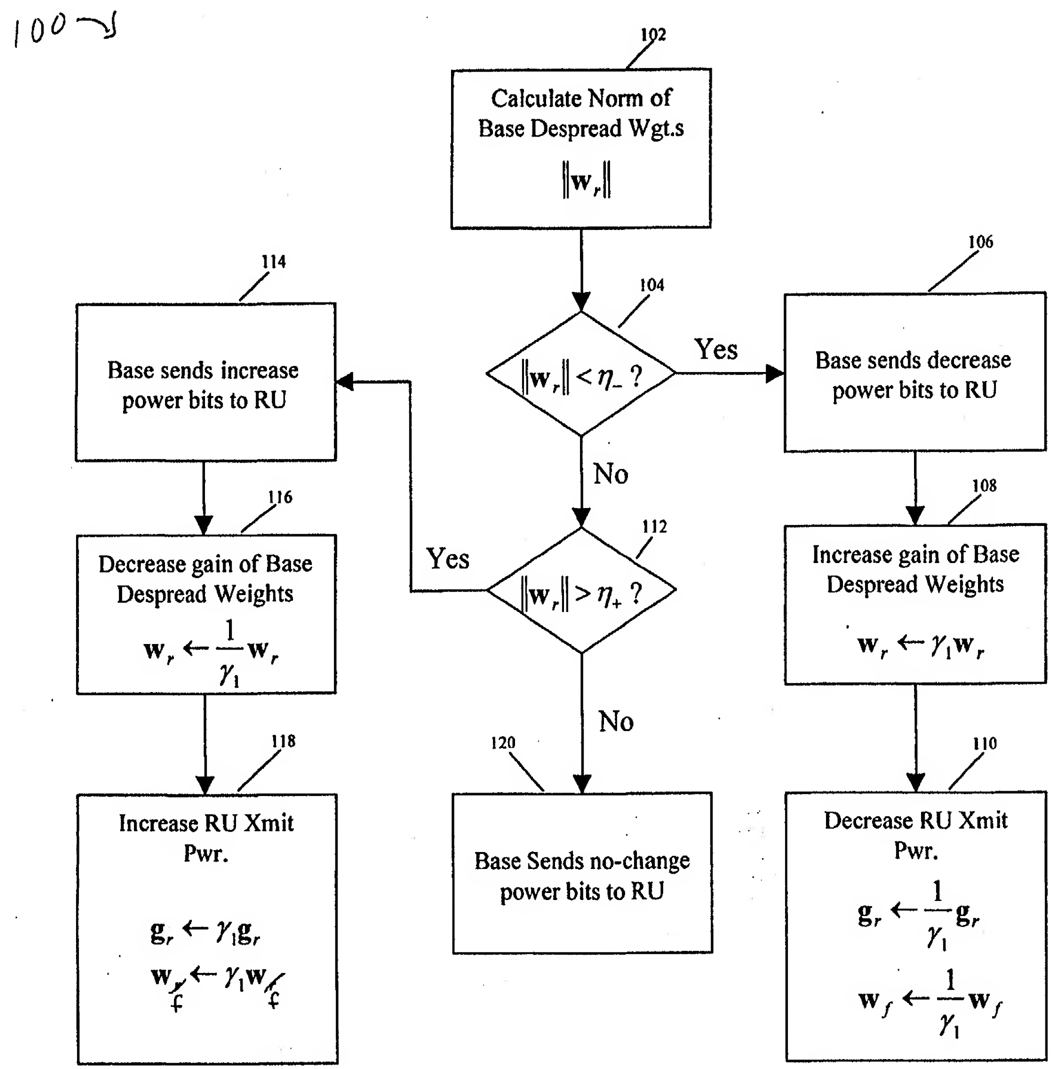

100
102
Calculate Norm of Base Despread Wgt.s
$\|\mathbf{w}_r\|$
104
$\|\mathbf{w}_r\| < \eta_- ?$
Yes
No
106
Base sends decrease power bits to RU
108
Increase gain of Base Despread Weights
$\mathbf{w}_r \leftarrow \gamma_1 \mathbf{w}_r$
110
Decrease RU Xmit Pwr.
$\mathbf{g}_r \leftarrow \frac{1}{\gamma_1} \mathbf{g}_r$
$\mathbf{w}_f \leftarrow \frac{1}{\gamma_1} \mathbf{w}_f$
112
$\|\mathbf{w}_r\| > \eta_+ ?$
Yes
No
114
Base sends increase power bits to RU
116
Decrease gain of Base Despread Weights
$\mathbf{w}_r \leftarrow \frac{1}{\gamma_1} \mathbf{w}_r$
118
Increase RU Xmit Pwr.
$\mathbf{g}_r \leftarrow \gamma_1 \mathbf{g}_r$
$\mathbf{w}_{\cancel{r}\,f} \leftarrow \gamma_1 \mathbf{w}_{\cancel{r}\,f}$
120
Base Sends no-change power bits to RU

FIG. ~~10~~ 8

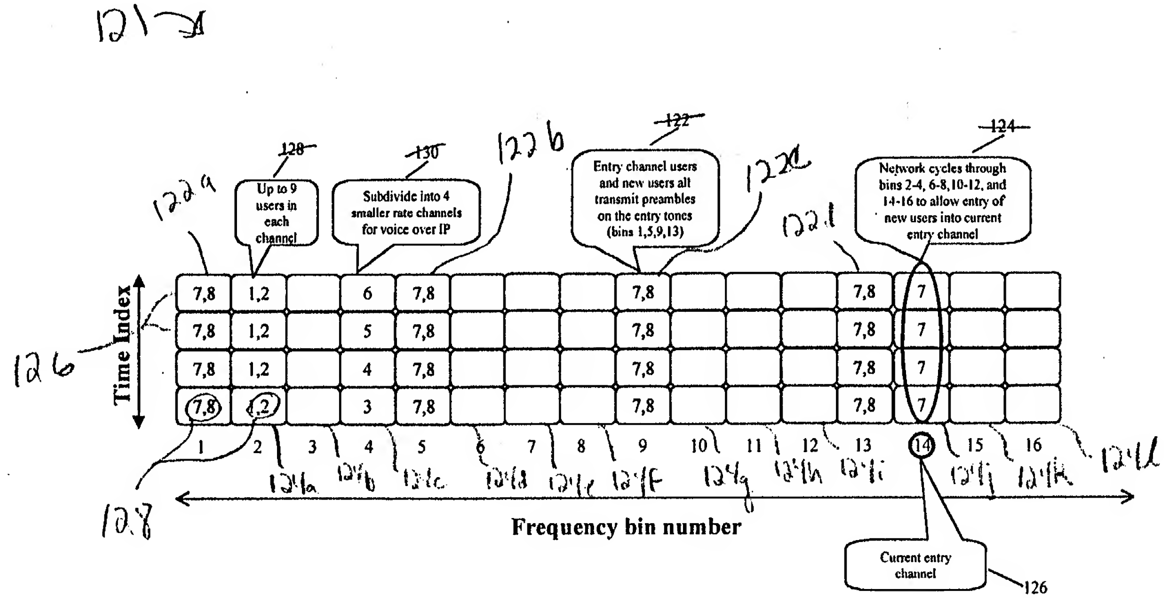

Up to 9 users in each channel
Subdivide into 4 smaller rate channels for voice over IP
Entry channel users and new users all transmit preambles on the entry tones (bins 1,5,9,13)
Network cycles through bins 2-4, 6-8, 10-12, and 14-16 to allow entry of new users into current entry channel
Time Index
Frequency bin number
Current entry channel
126

FIG. ~~11~~ 9

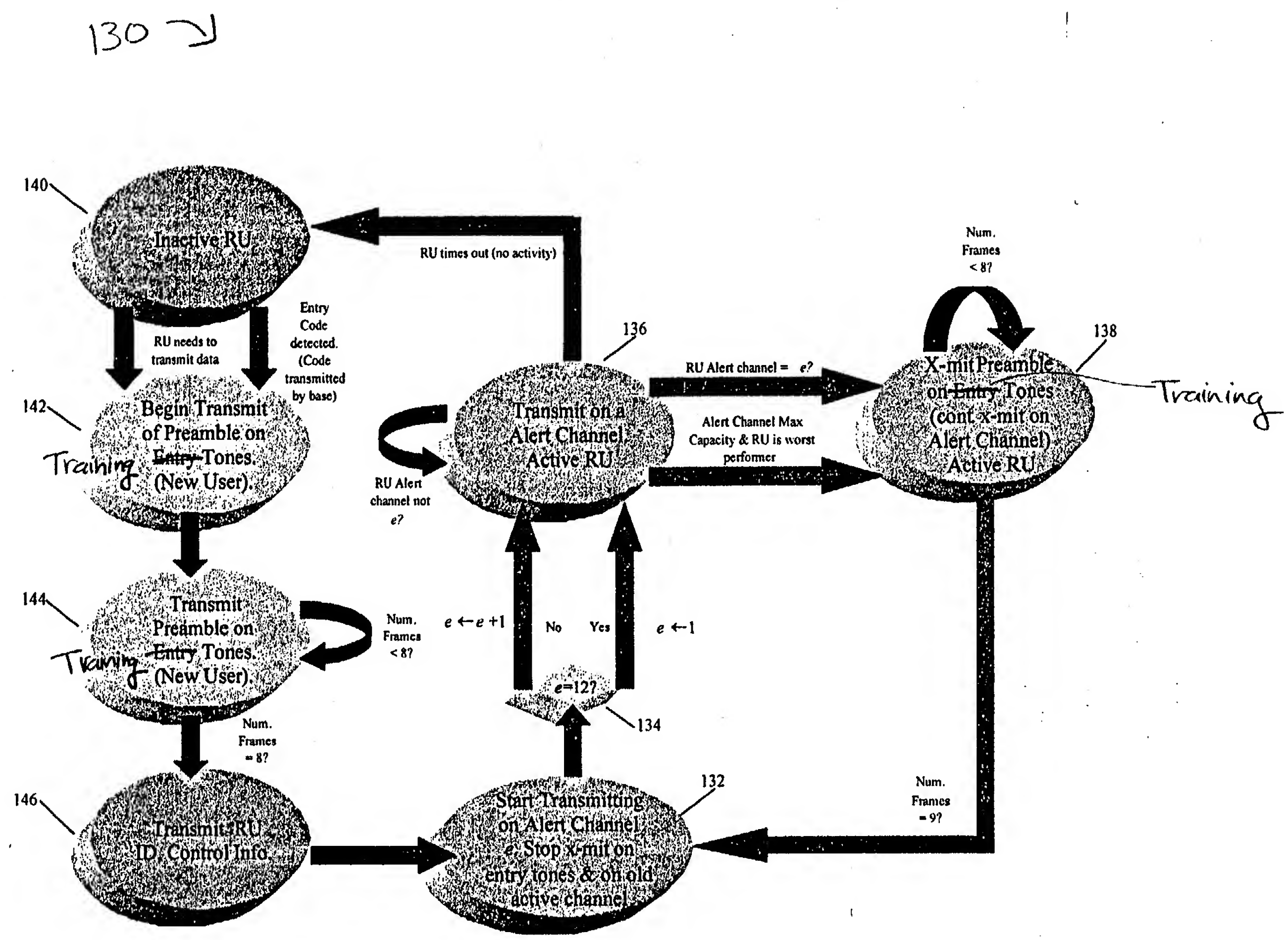

130
140
Inactive RU
RU needs to transmit data
Entry Code detected. (Code transmitted by base)
142
Begin Transmit of Preamble on ~~Entry~~ Tones. (New User)
Training
144
Transmit Preamble on ~~Entry~~ Tones. (New User)
Training
Num. Frames < 8?
Num. Frames = 8?
146
Transmit RU ID, Control Info.
RU times out (no activity)
136
Transmit on a Alert Channel Active RU
RU Alert channel not e?
RU Alert channel = e?
Alert Channel Max Capacity & RU is worst performer
138
X-mit Preamble on ~~Entry~~ Tones (cont x-mit on Alert Channel) Active RU
Training
Num. Frames < 8?
Num. Frames = 9?
e ← e +1
No
Yes
e ← 1
e=12?
134
132
Start Transmitting on Alert Channel e. Stop x-mit on entry tones & on old active channel

FIG. ~~12~~ 10

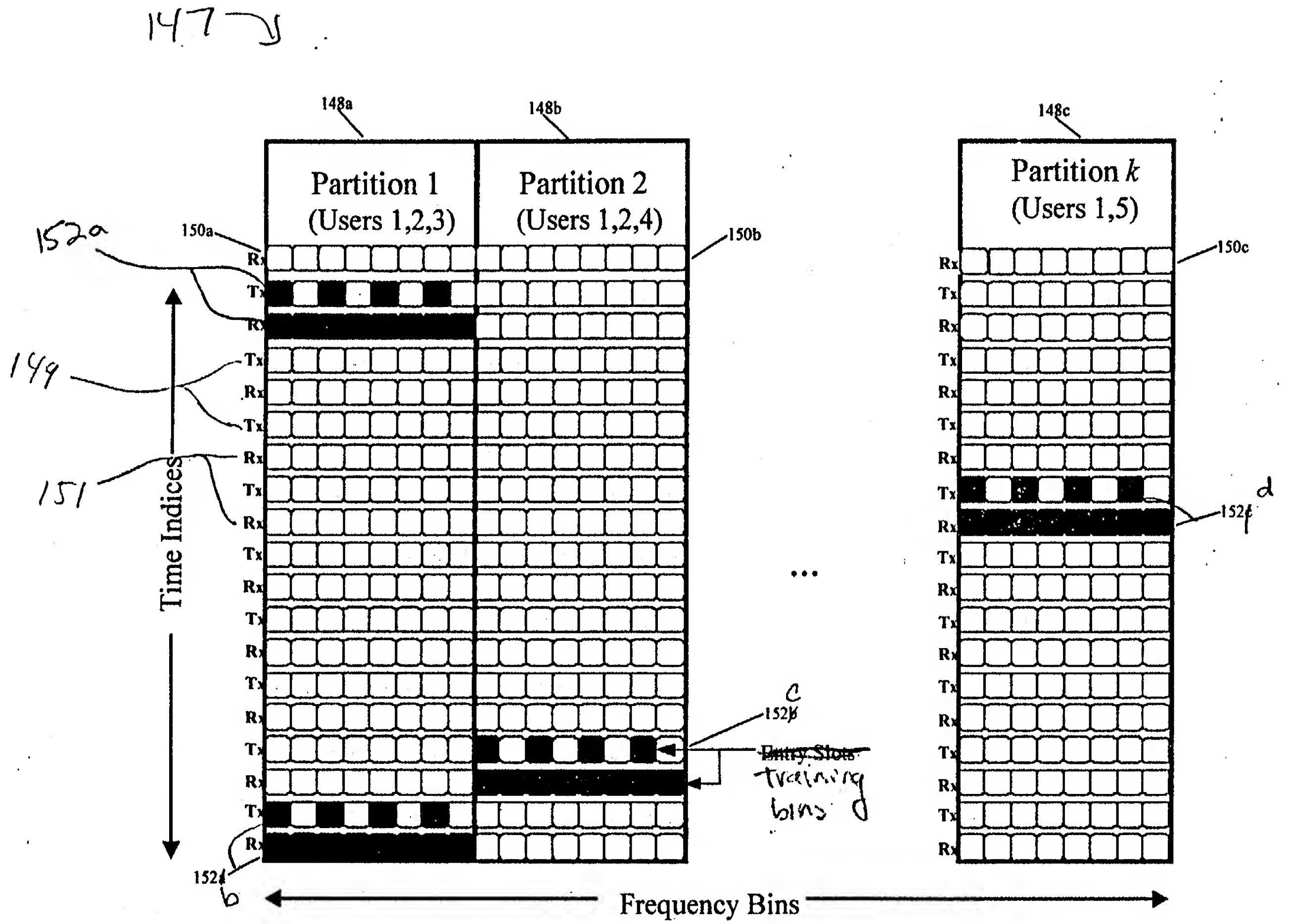

147
148a
148b
148c
Partition 1
(Users 1,2,3)
Partition 2
(Users 1,2,4)
Partition k
(Users 1,5)
150a
150b
150c
152a
149
151
152b
152c
152d
b
c
d
Rx
Tx
Time Indices
~~Entry Slots~~
training bins
Frequency Bins

FIG. ~~13~~ 11

153

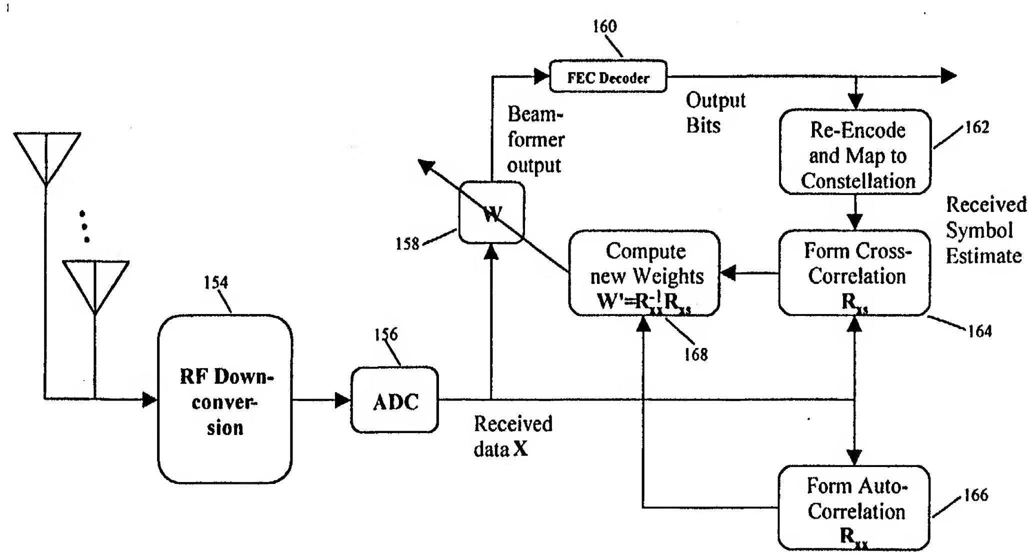
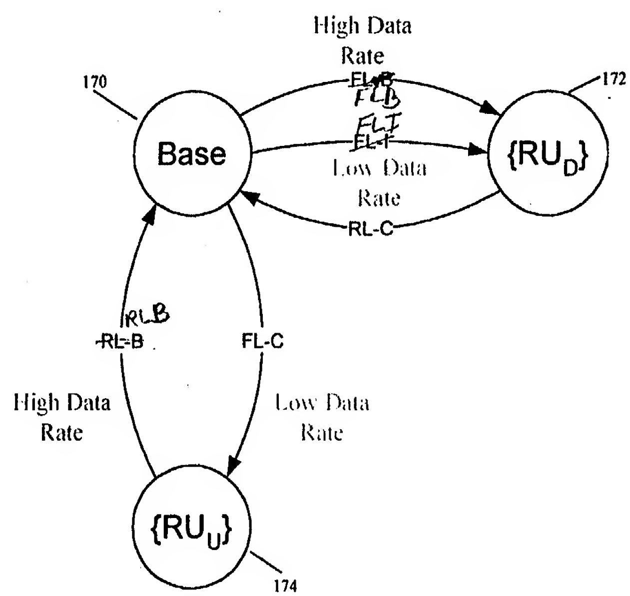
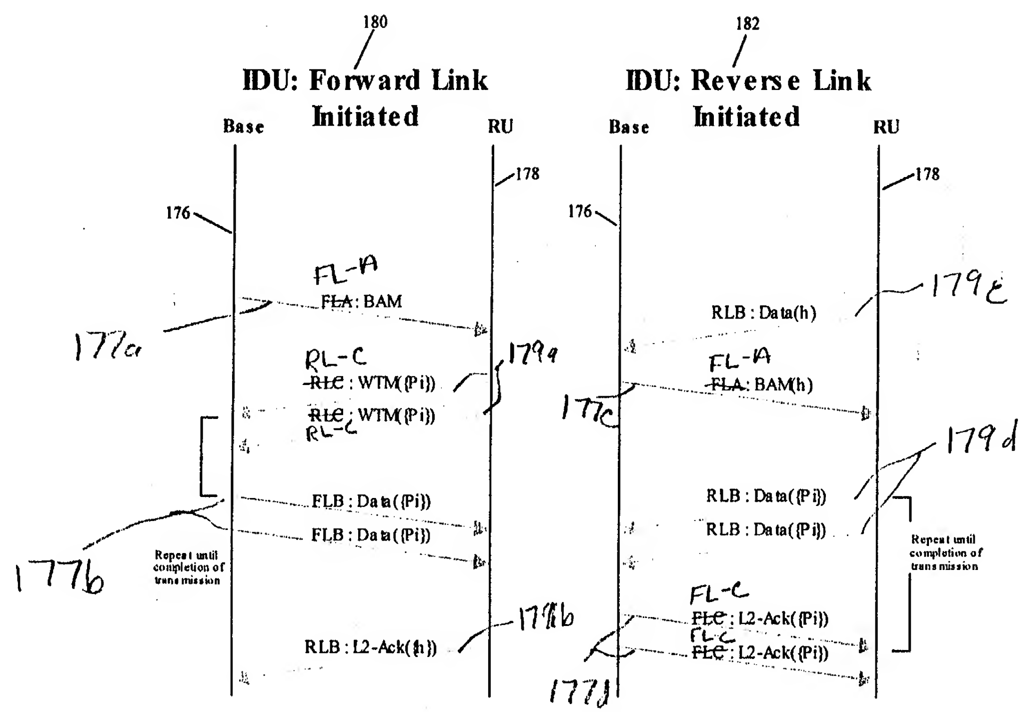

160
FEC Decoder
Output
Bits
Beam-
former
output
Re-Encode
and Map to
Constellation
162
Received
Symbol
Estimate
W
158
Compute
new Weights
$W'=R_{xx}^{-1}R_{xs}$
Form Cross-
Correlation
R_{xs}
164
168
154
RF Down-
conver-
sion
156
ADC
Received
data X
Form Auto-
Correlation
R_{xx}
166

FIG. ~~14~~ 12

169


High Data
Rate
~~FL-B~~
FLD
FLI
~~FL-I~~
Low Data
Rate
RL-C
170
Base
172
{RU_D}
RLB
~~RL-B~~
High Data
Rate
FL-C
Low Data
Rate
{RU_U}
174

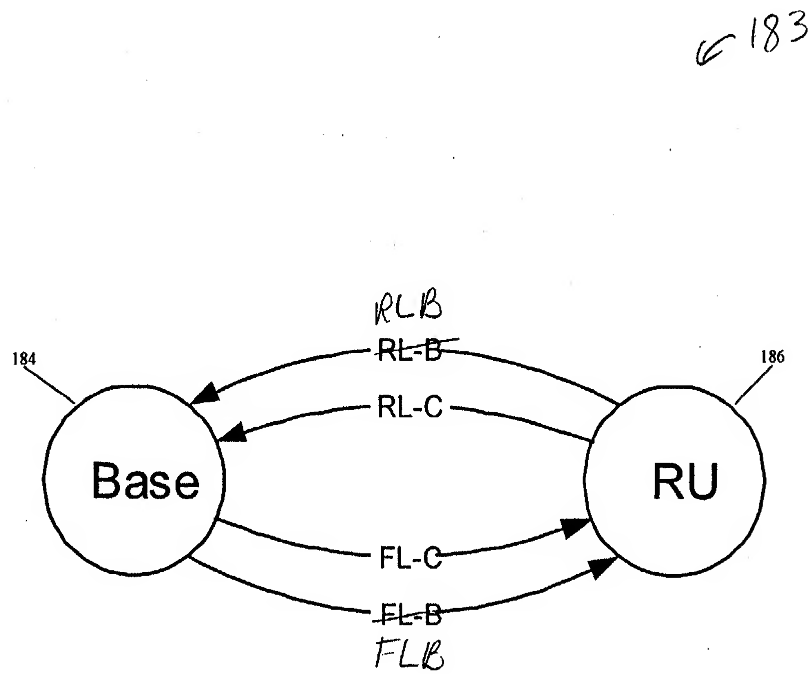
13
FIG. ~~15~~



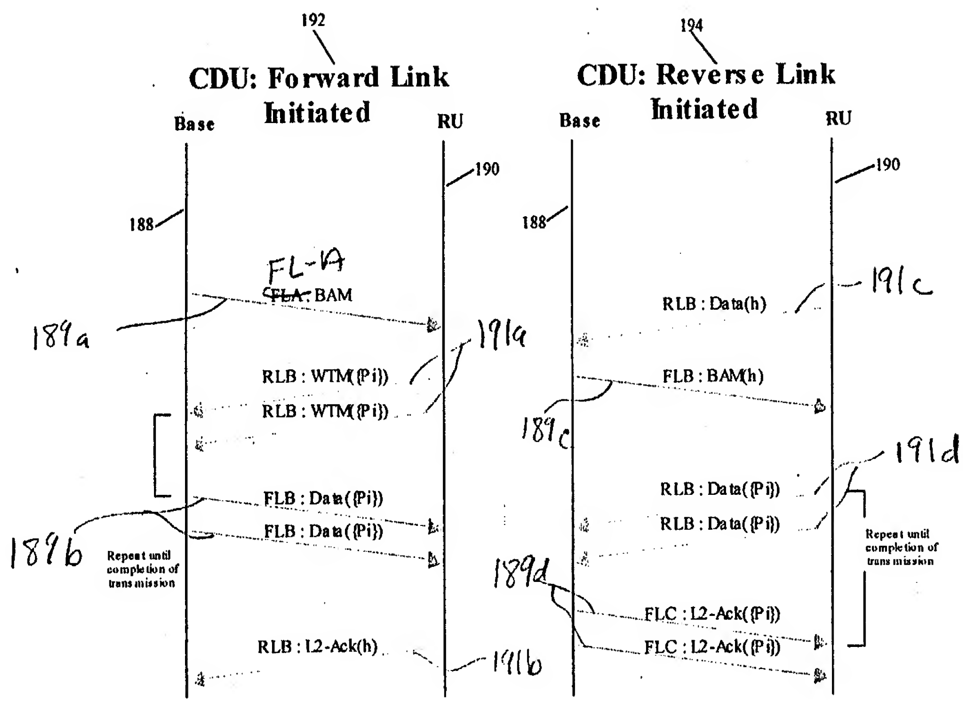
180
IDU: Forward Link Initiated
Base
RU
176
178
FL-A
~~FLA~~: BAM
177a
RL-C
~~RLC~~: WTM(Pi))
~~RLC~~: WTM(Pi))
RL-C
179a
FLB : Data((Pi))
FLB : Data((Pi))
177b
Repeat until completion of transmission
RLB : L2-Ack((h))
179b
182
IDU: Reverse Link Initiated
Base
RU
176
178
RLB : Data(h)
179e
FL-A
~~FLA~~: BAM(h)
177c
179d
RLB : Data((Pi))
RLB : Data((Pi))
Repeat until completion of transmission
FL-C
~~FLC~~: L2-Ack((Pi))
FL-C
~~FLC~~: L2-Ack((Pi))
177d

14

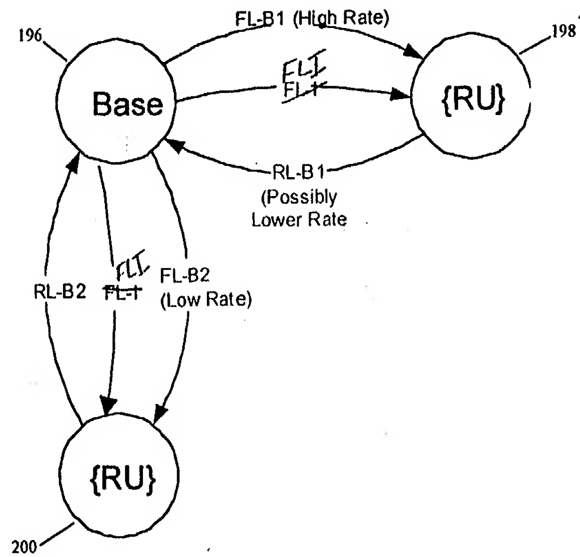
FIG. ~~16~~



15
FIG. ~~17~~

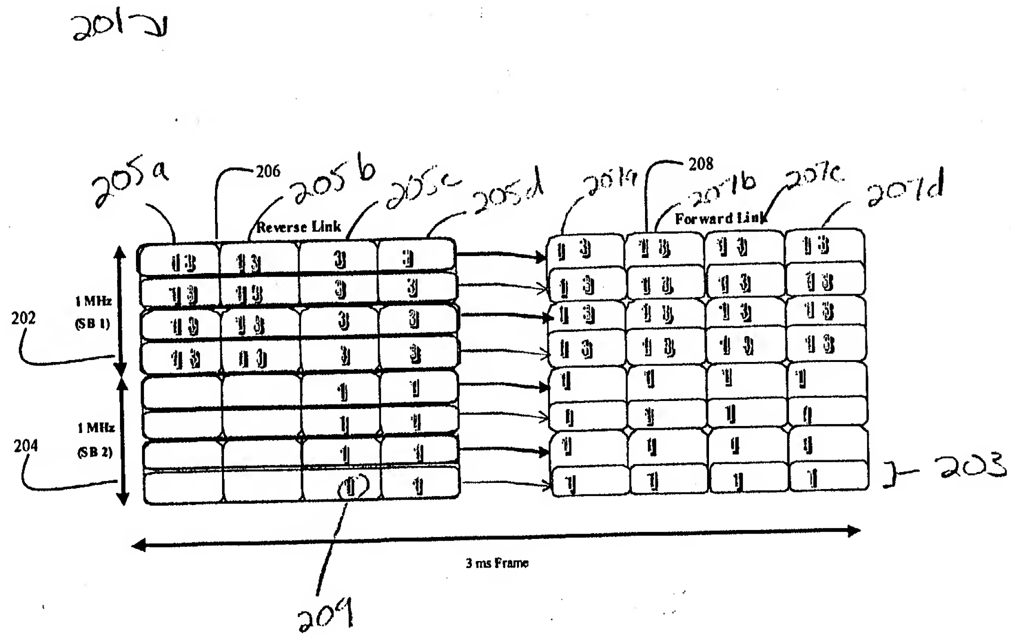

192
CDU: Forward Link Initiated
Base
RU
190
188
FL-1A
~~FLA:~~ BAM
189a
191a
RLB : WTM(Pi))
RLB : WTM(Pi))
FLB : Data(Pi))
FLB : Data(Pi))
189b
Repeat until completion of transmission
RLB : L2-Ack(h)
191b
194
CDU: Reverse Link Initiated
Base
RU
190
188
RLB : Data(h)
191c
FLB : BAM(h)
189c
191d
RLB : Data(Pi))
RLB : Data(Pi))
Repeat until completion of transmission
189d
FLC : L2-Ack(Pi))
FLC : L2-Ack(Pi))

16
FIG. ~~18~~



196
198
200
Base
{RU}
{RU}
FL-B1 (High Rate)
FLI
~~FL-1~~
RL-B1
(Possibly
Lower Rate
RL-B2
FLI
~~FL-1~~
FL-B2
(Low Rate)

FIG. ~~19~~ 17


201
202
204
1 MHz (SB 1)
1 MHz (SB 2)
205a
205b
205c
205d
206
Reverse Link
207a
207b
207c
207d
208
Forward Link
203
209
3 ms Frame

18

FIG. 20

212a

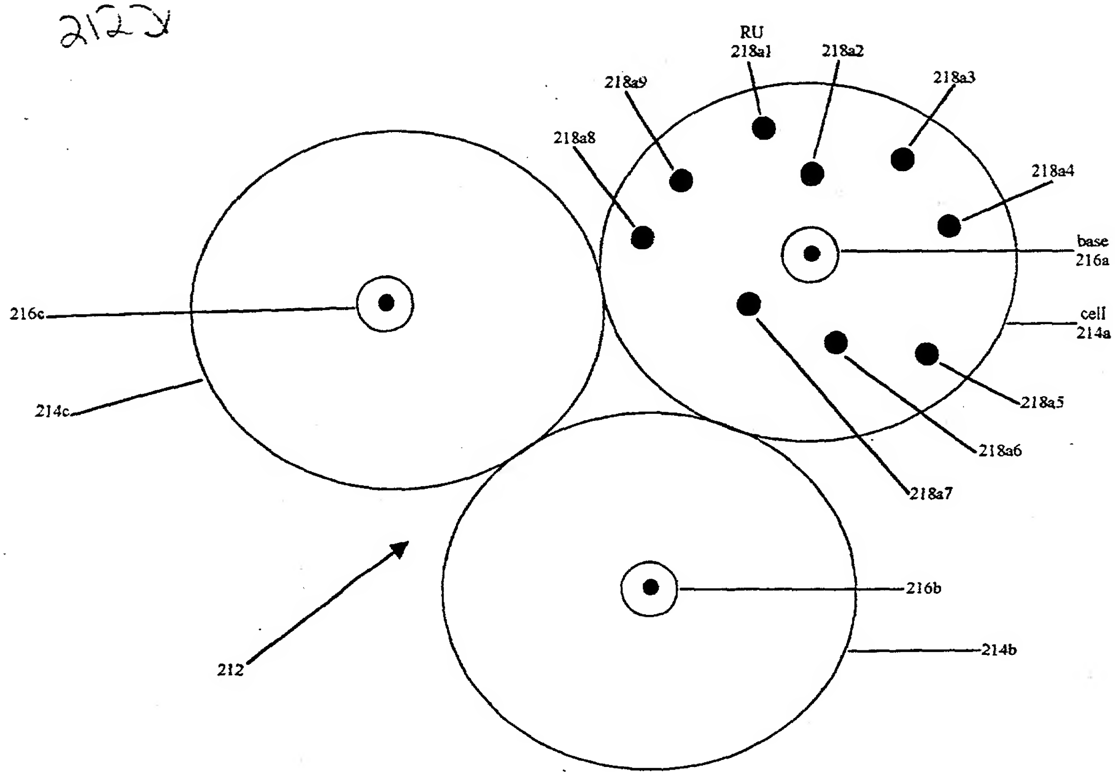


Figure 19. ~~Network with base units and RUs~~.

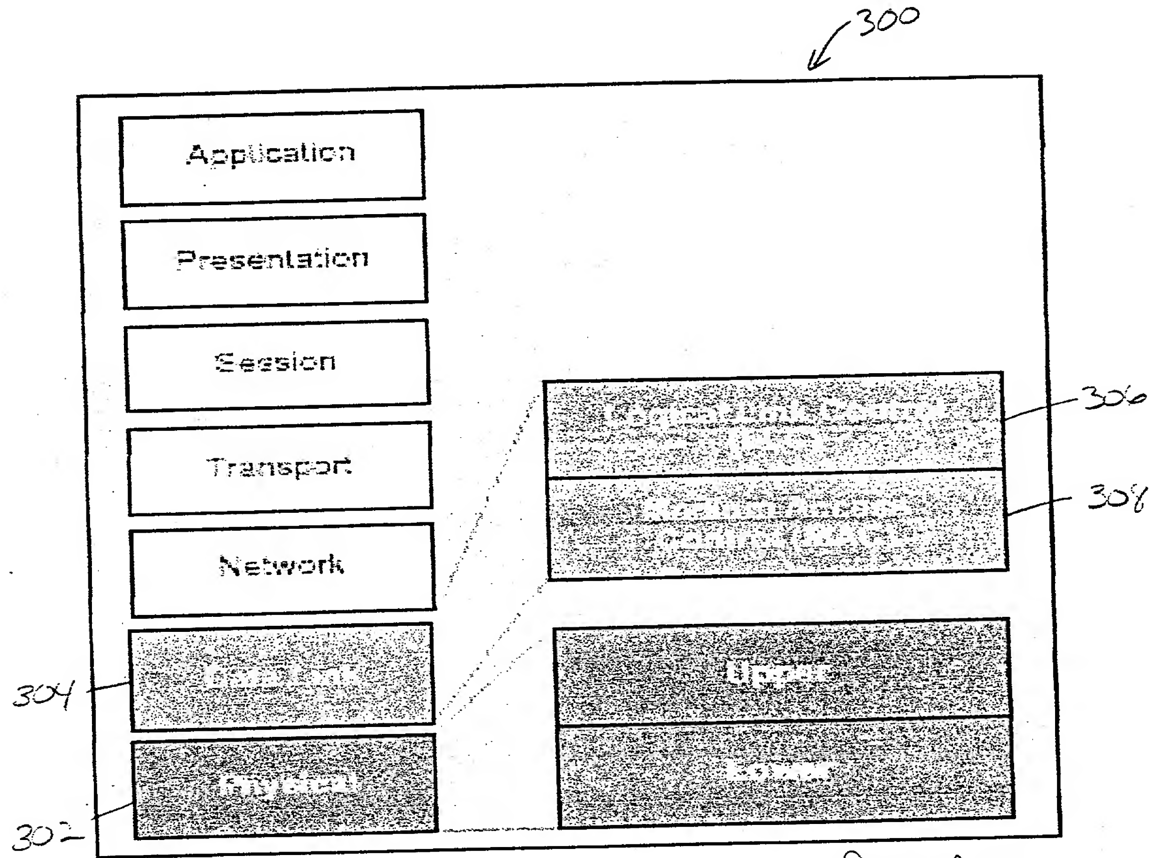


Figure 20. ~~OSI Protocol Framework~~ Prior Art

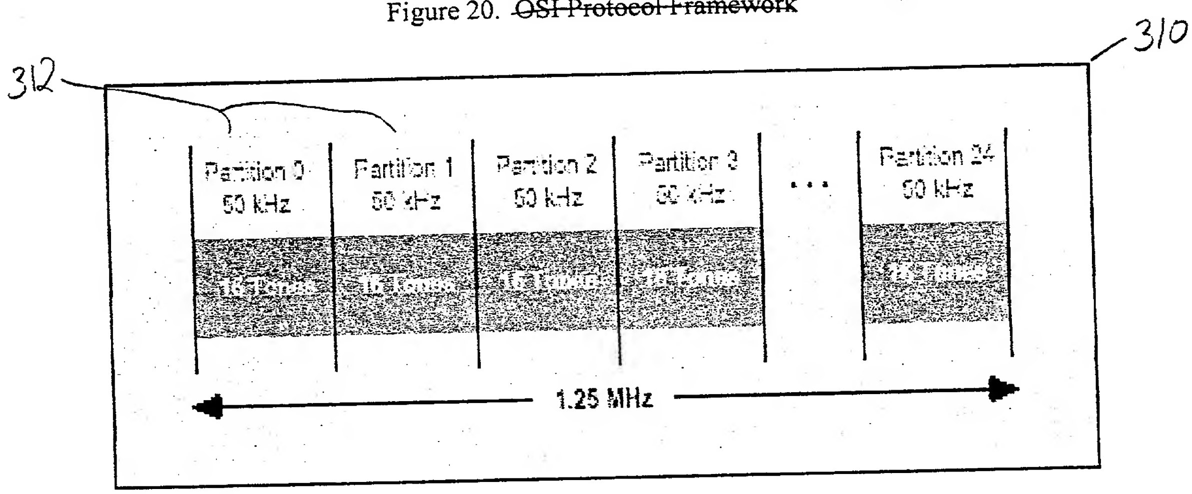


Figure 21. ~~Tone Partitions within a~~ Subband

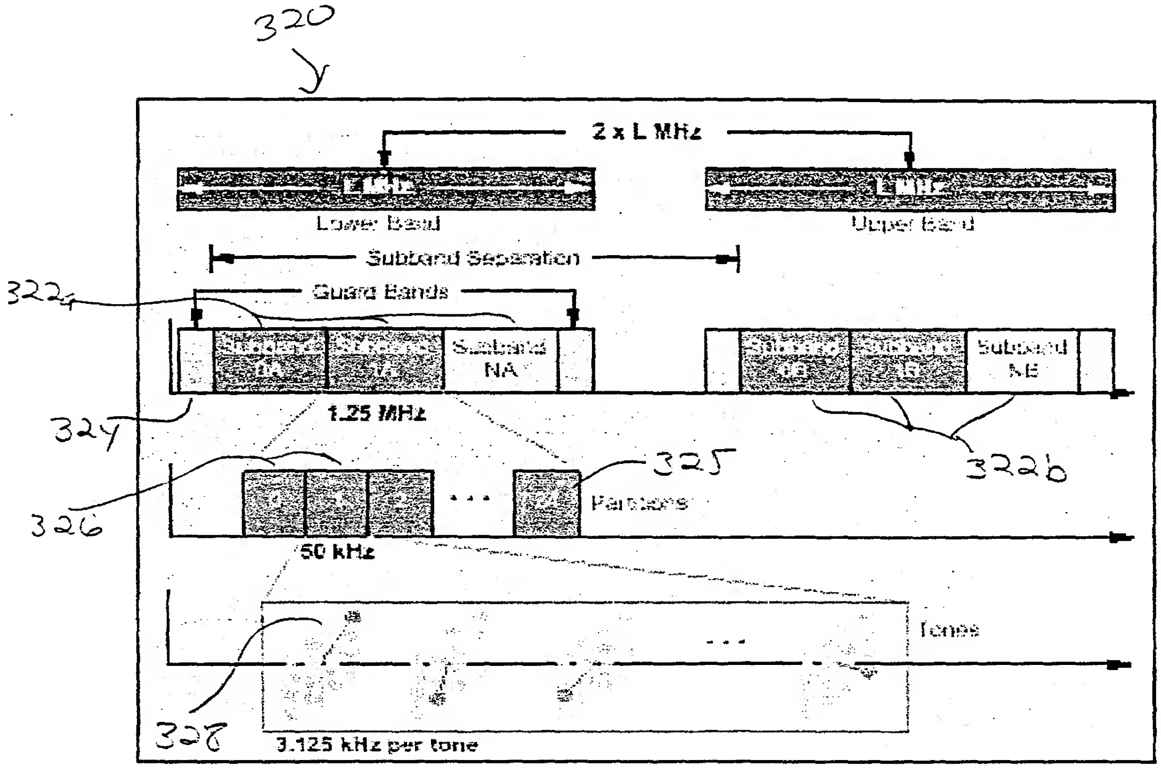


Figure 22. ~~Frequency plan with a spectral spreading factor of two.~~

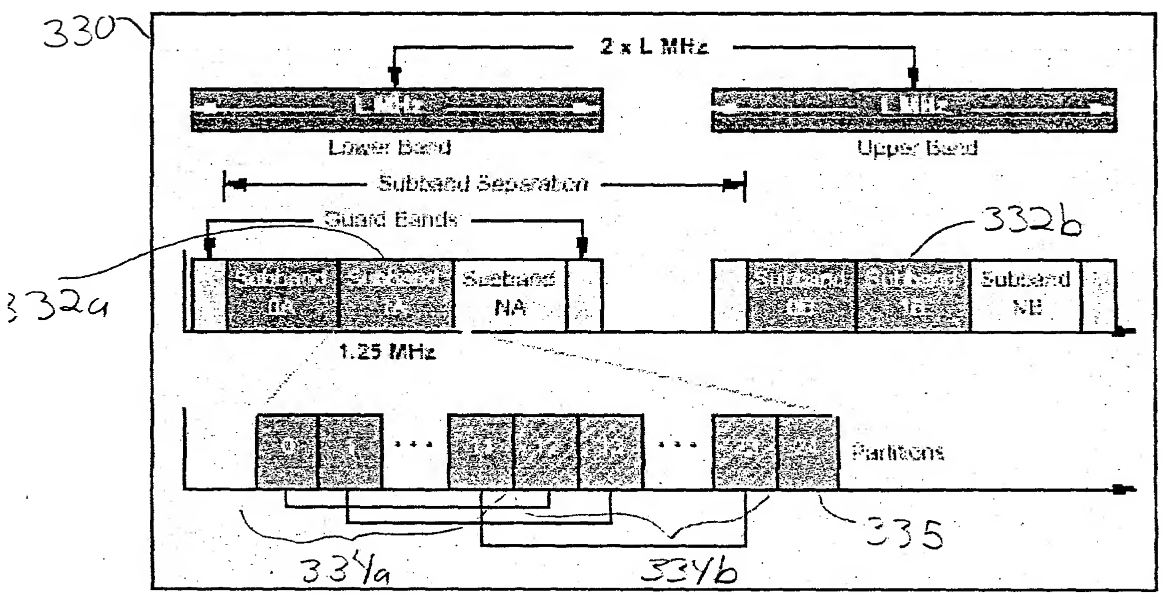


Figure 23. ~~Frequency plan with a spreading factor of four.~~

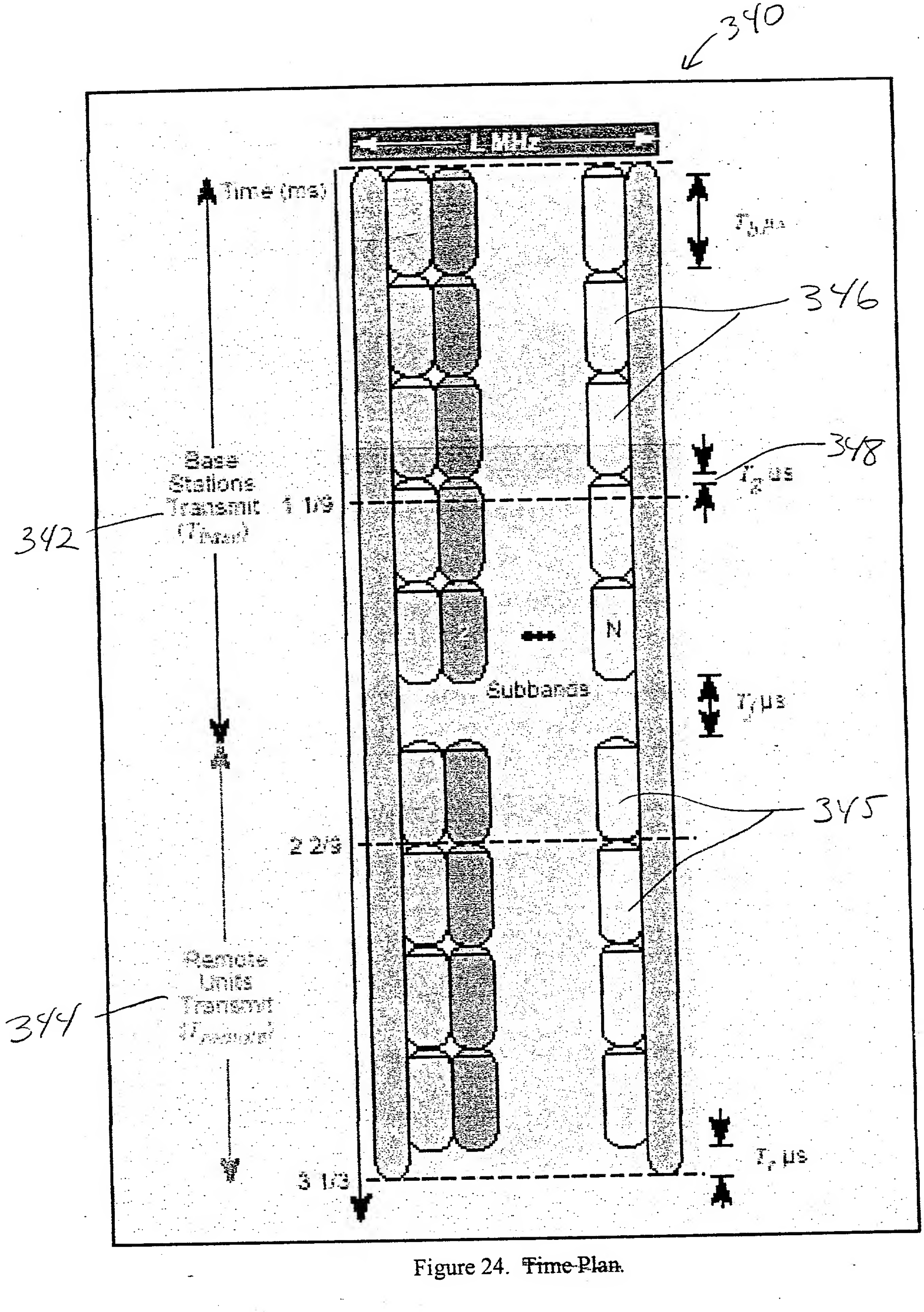


Figure 24. ~~Time Plan.~~

350

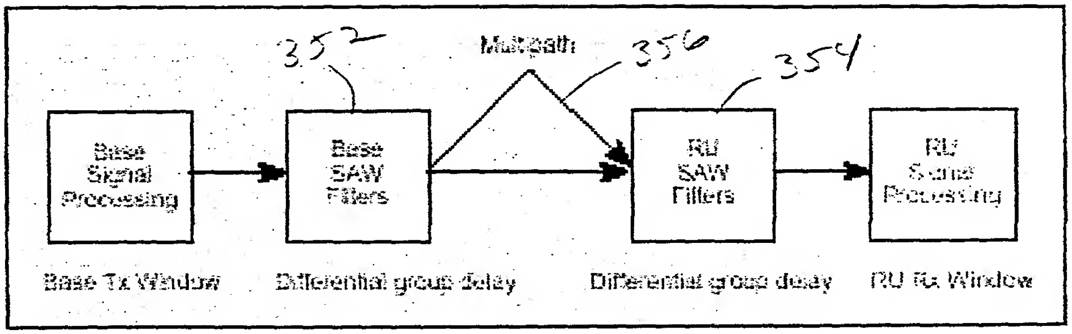


Figure 25. ~~Guard Time Factors~~

360

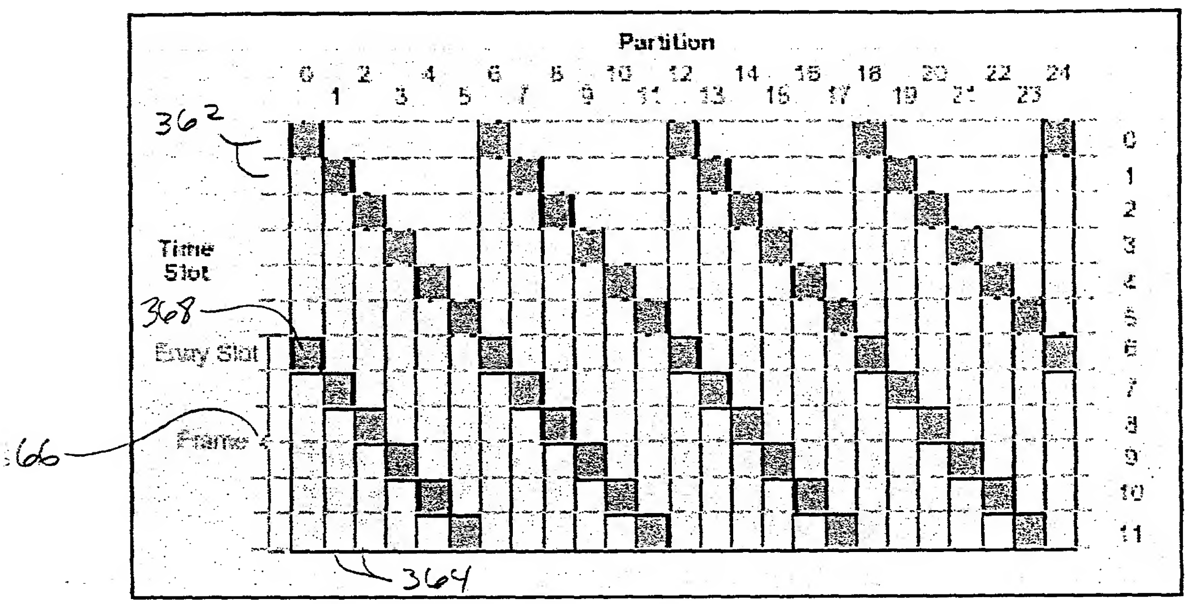


Figure 26. ~~Partition layout for a lower subband with a spectral spreading factor of 2; 4 x 6 + 1~~ layout; six time slots per frame.

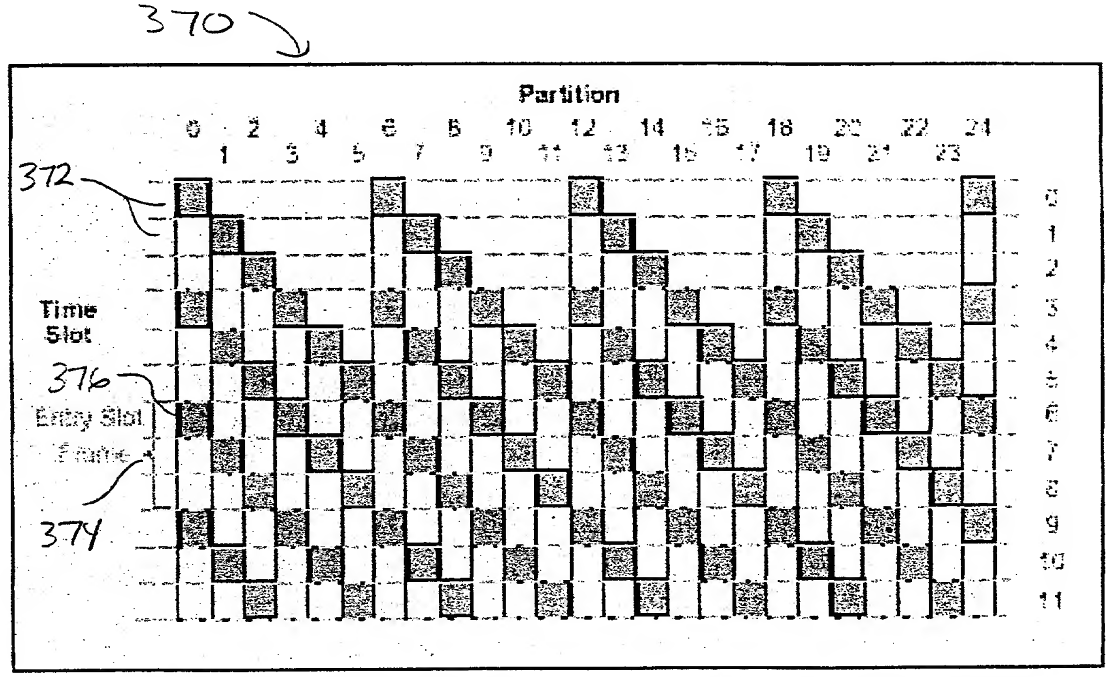


Figure 27. ~~Partition layout for a lower~~ subband with a spectral spreading factor of 2; 8 x 3 + 1 layout; three timeslots per frame.

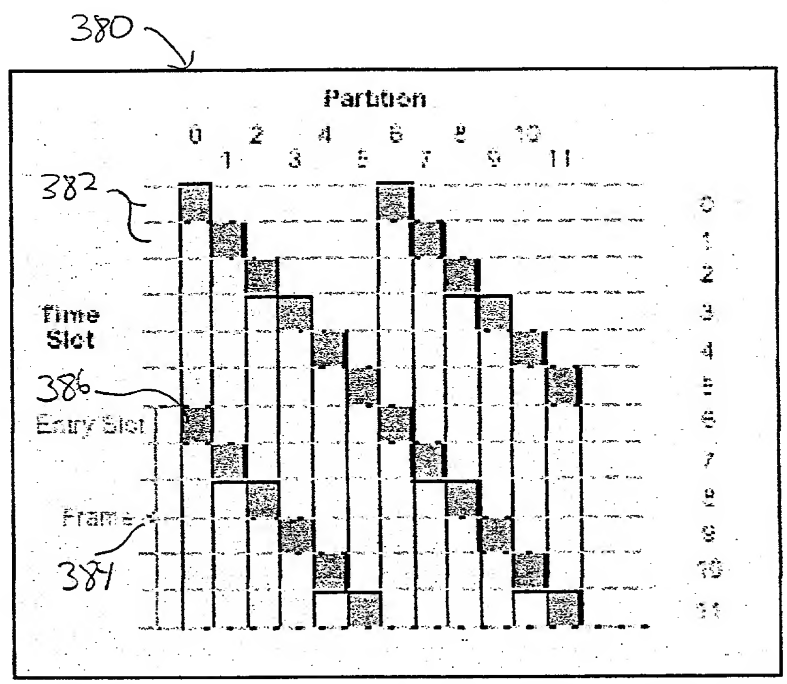


Figure 28. ~~Partition layout for a~~ lower subband with a spectral spreading factor of 4; a 2 x 6 layout; six time slots per frame.

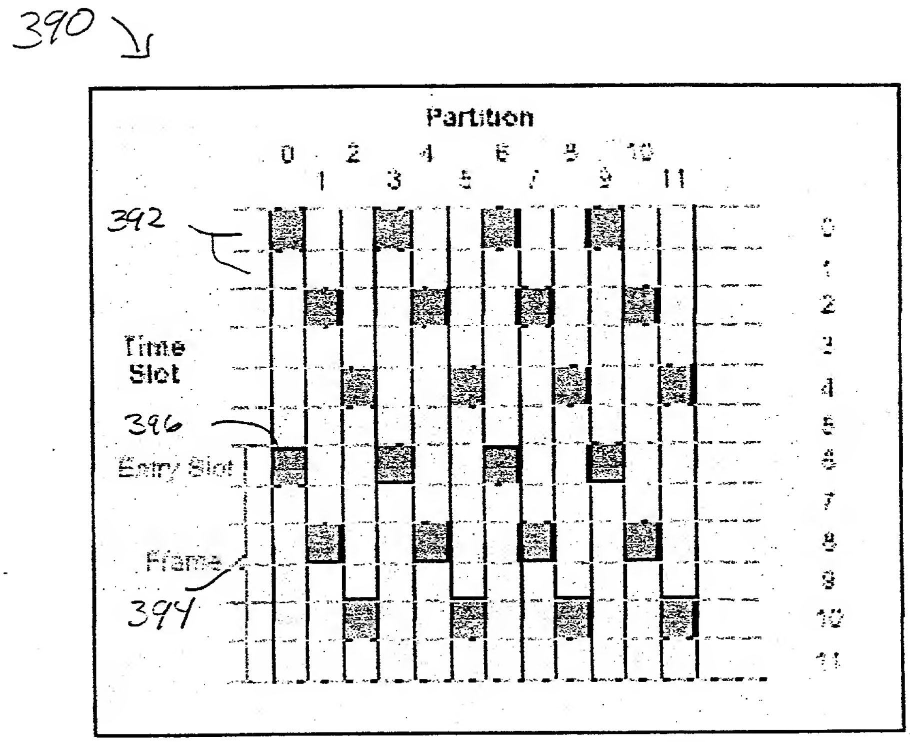


Figure 29. ~~Partition layout for a lower subband with a spectral spreading factor of 4, 2 × 6 layout, six time slots per frame.~~

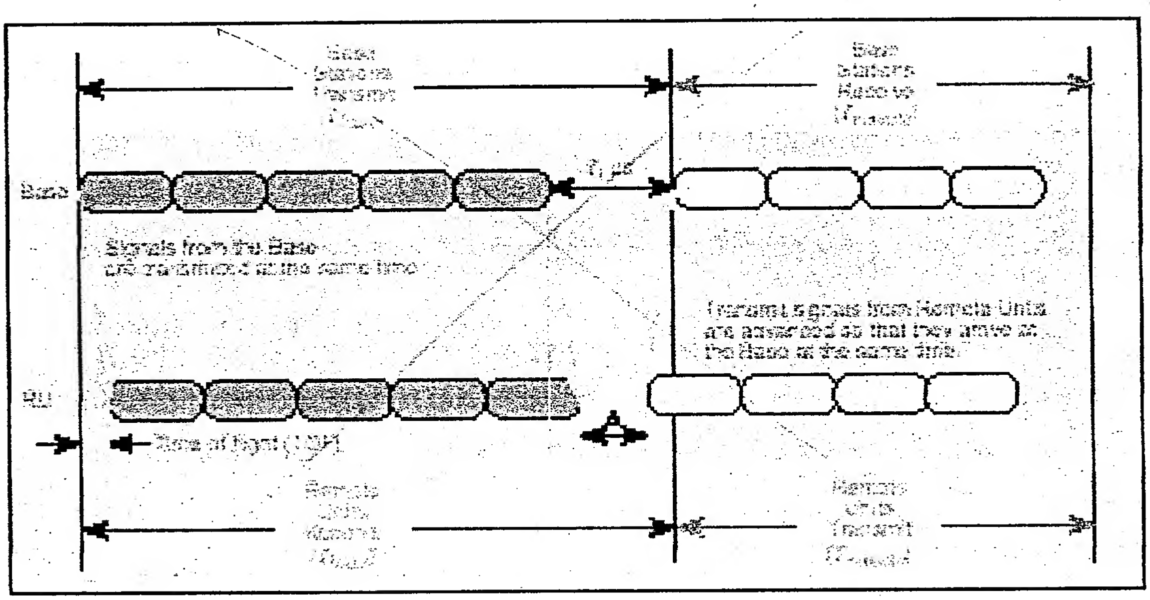


Figure 30. Cell Radius Constraint.

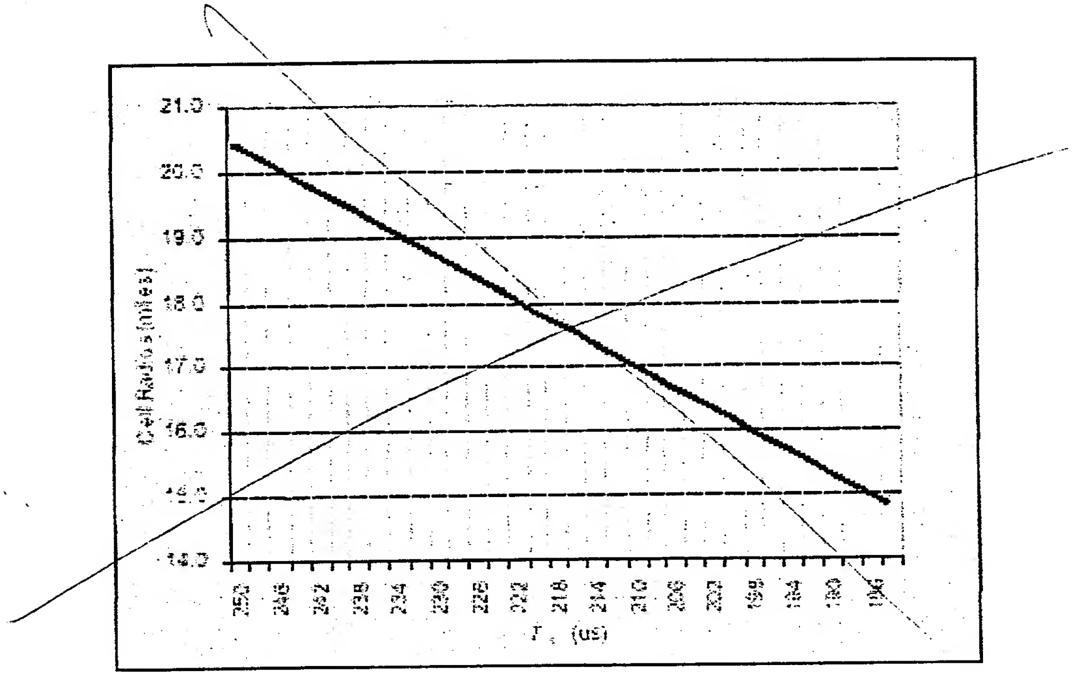


Figure 31. Cell Radius as a function of the excess forward link time, and the processing time.

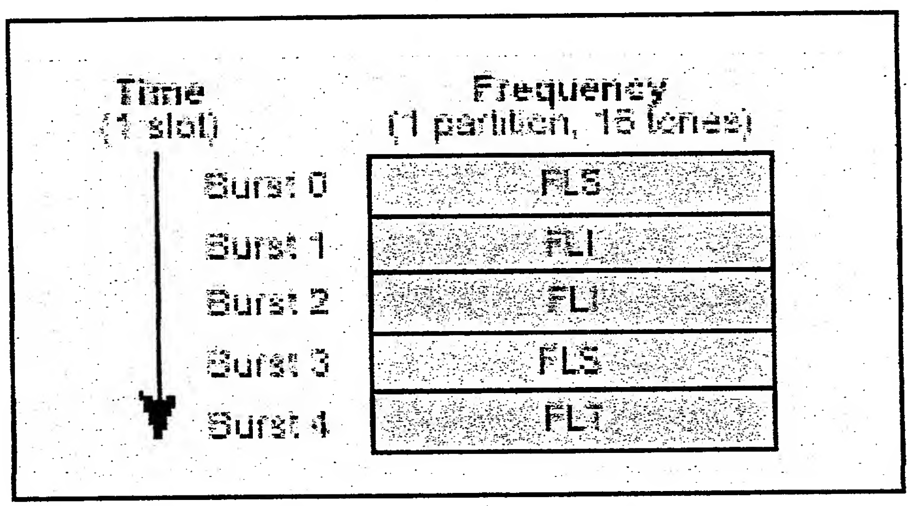


Figure ~~32. Burst Assignments in the Forward Link Entry Slot~~.
30

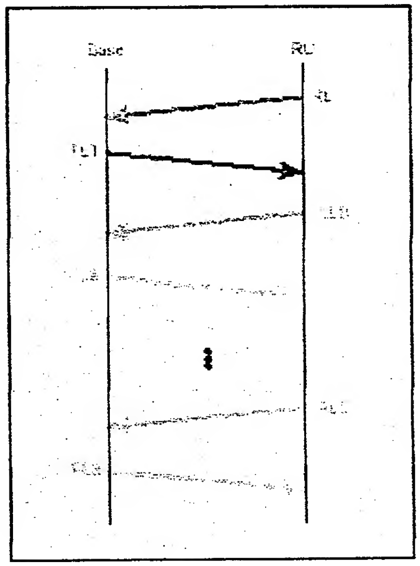


Figure ~~33~~. ~~Channel Structure.~~
31

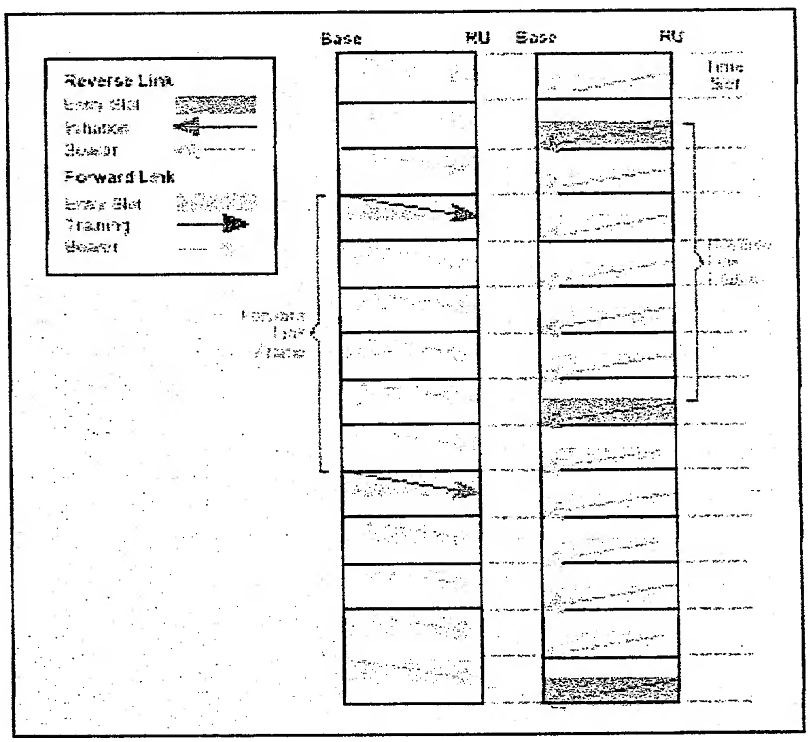


Figure ~~34~~. ~~Frame Offset.~~
32

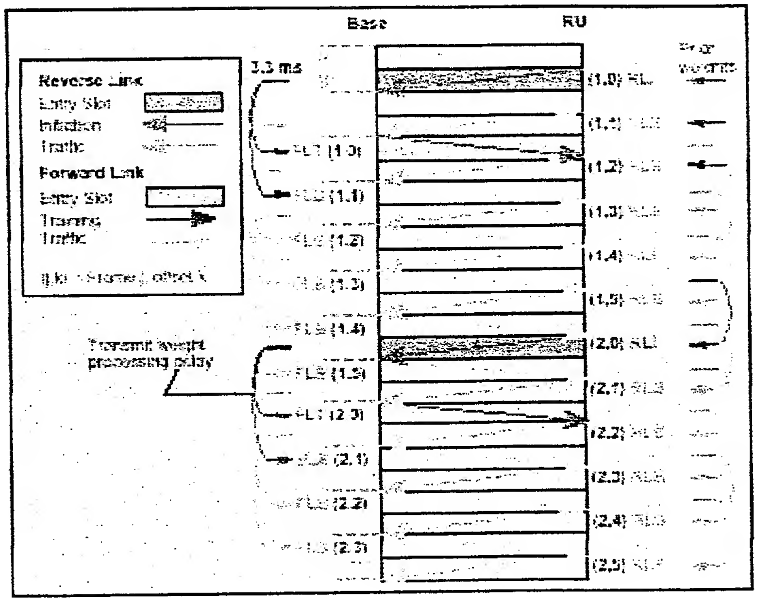


Figure ~~35~~ 33. ~~Reverse Link Initiated Transfer~~.

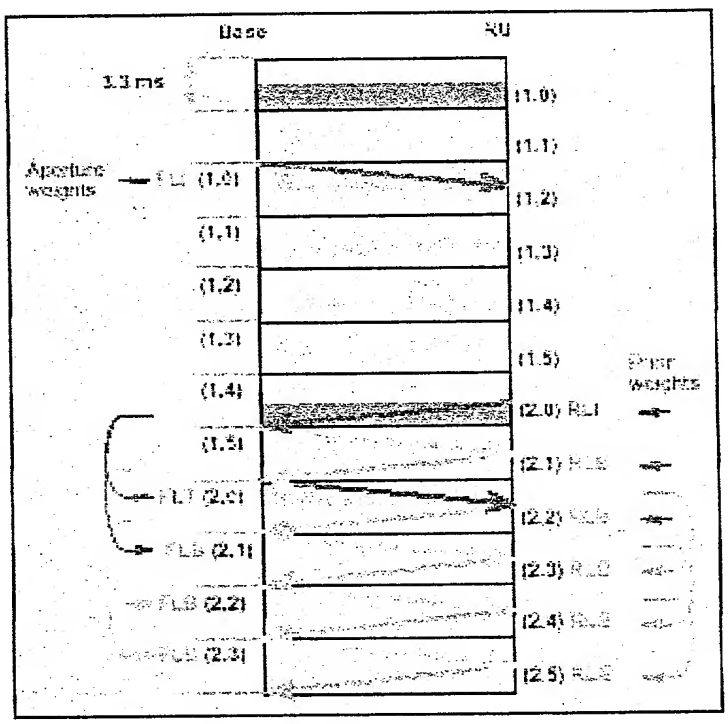


Figure ~~36~~ 34. ~~Forward Link Initiated Transfer~~.

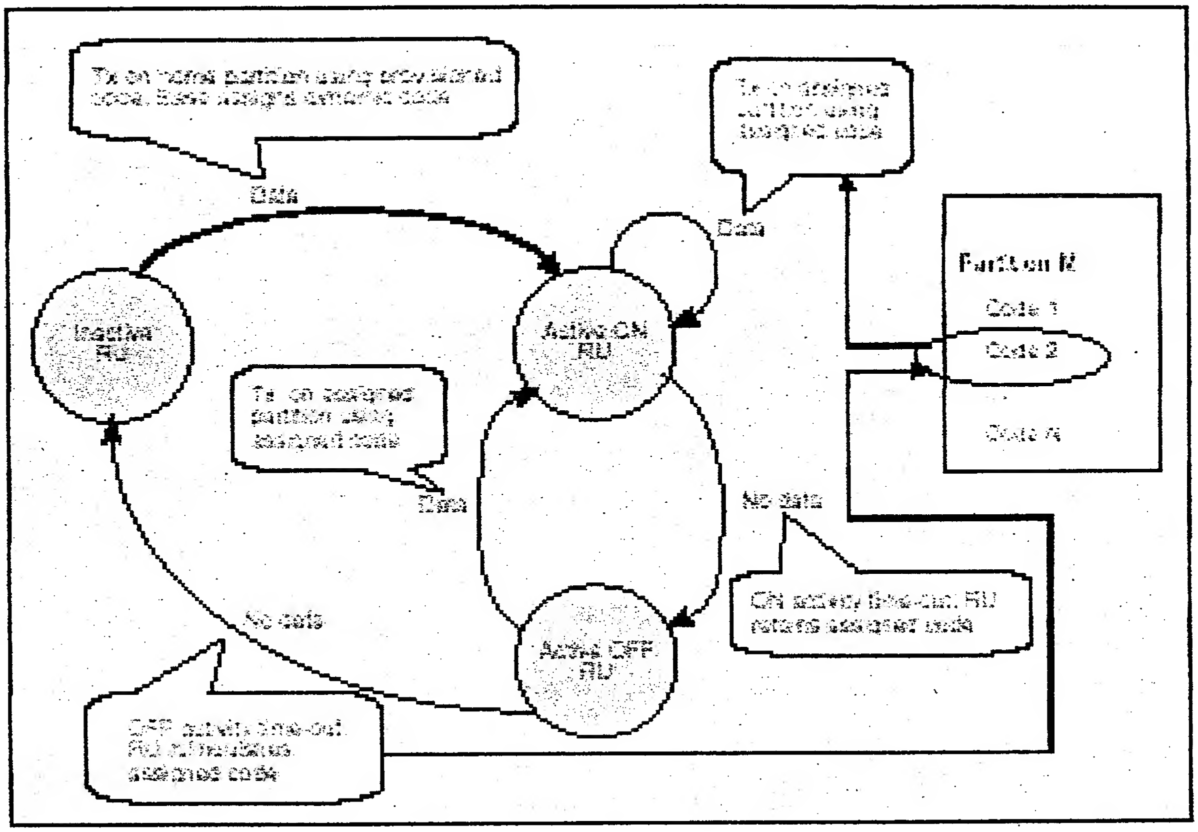


Figure ~~37~~ 5. ~~Dynamic RLI Code Assignment.~~

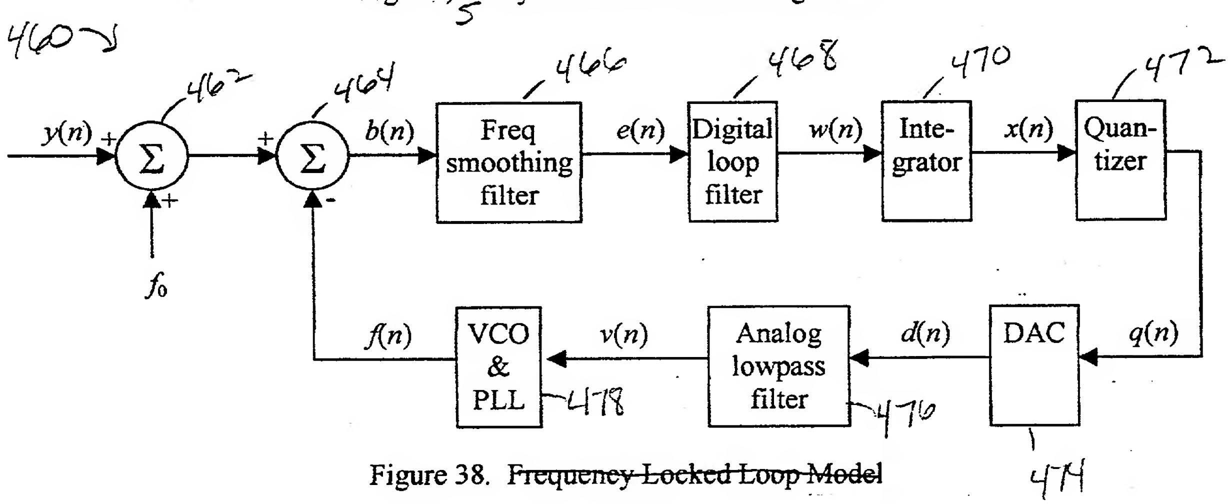


Figure 38. ~~Frequency Locked Loop Model~~

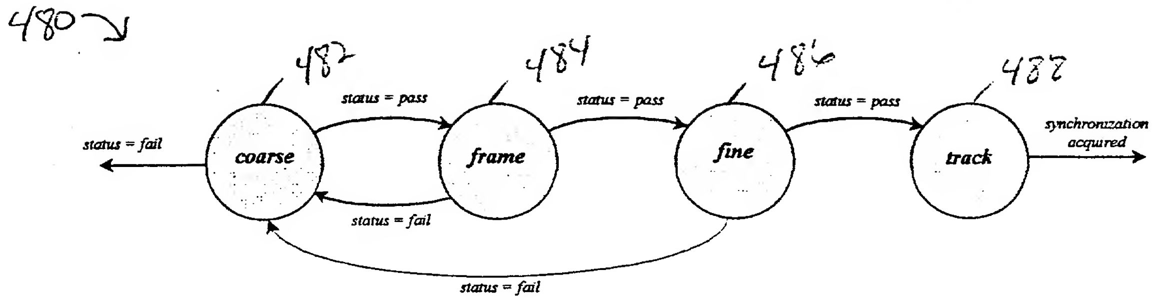


Figure 39. Synchronization Acquisition State Diagram.

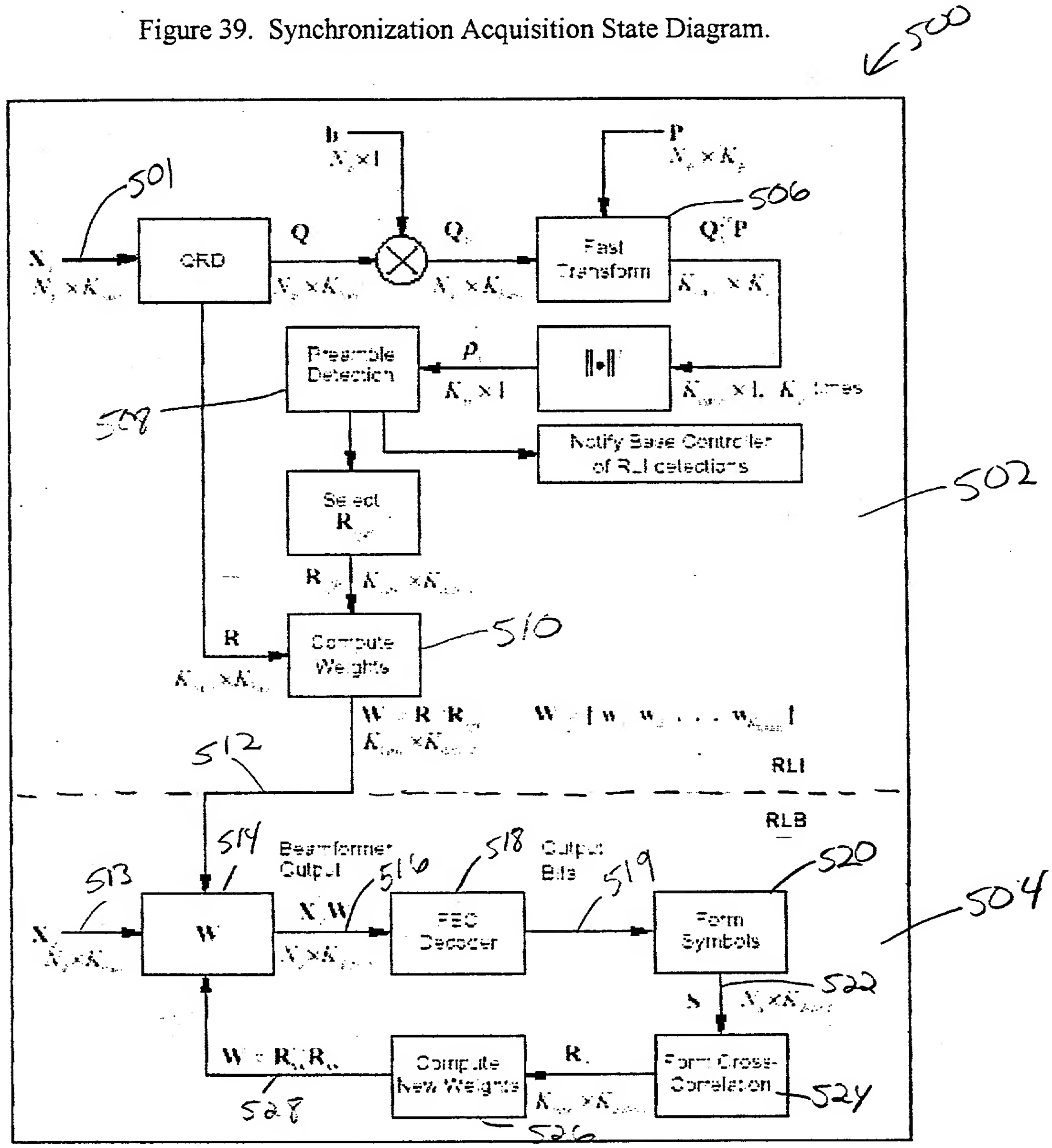


Figure ~~40~~ 39. Base Receiver Block Diagram.

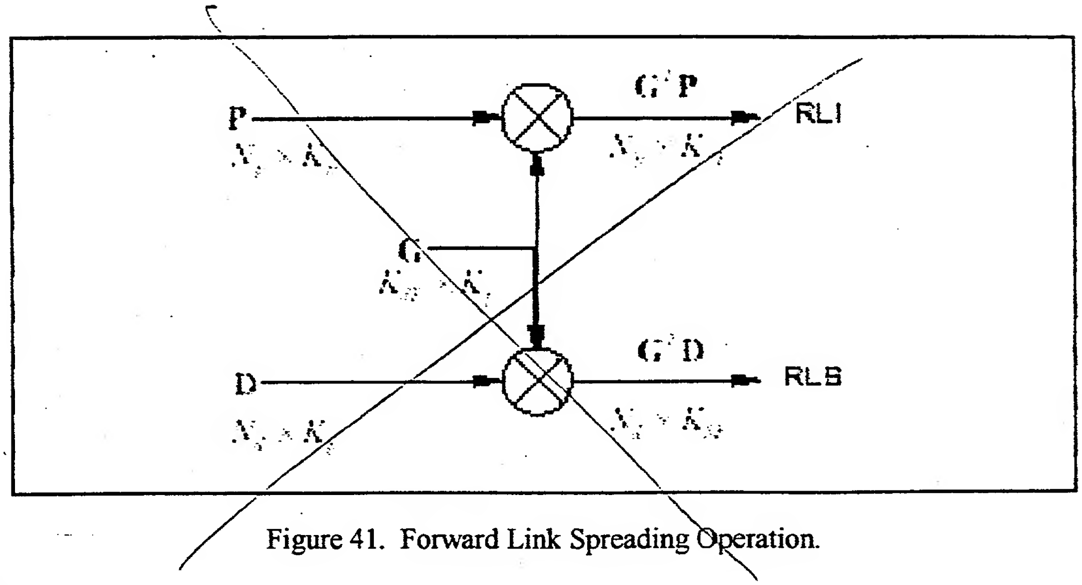


Figure 41. Forward Link Spreading Operation.

530

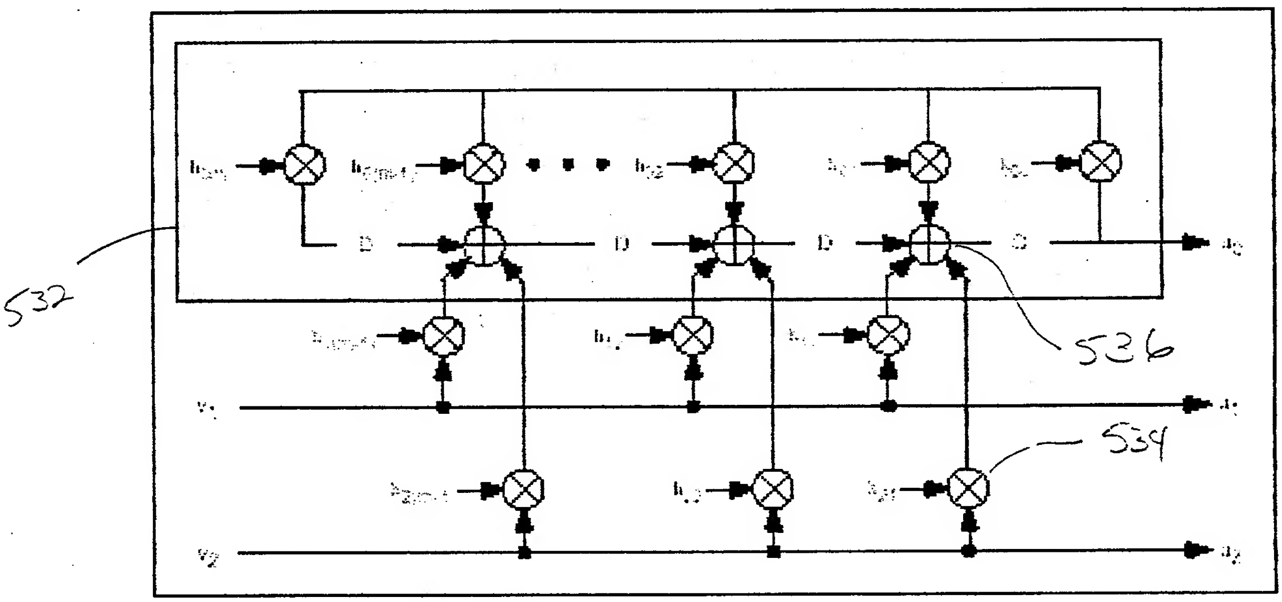


Figure ~~42~~. ~~Rate 2/3, 2^m-state Convolutional Encoder with Feedback~~.

39

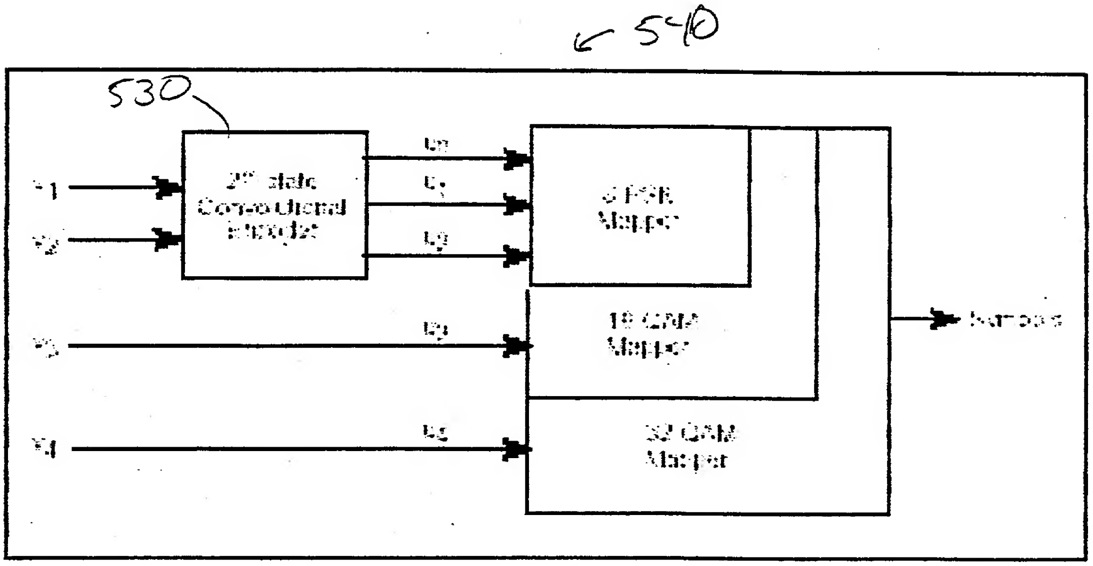


Figure ~~43~~ 40. ~~Scalable Trellis-Coded Mapper.~~

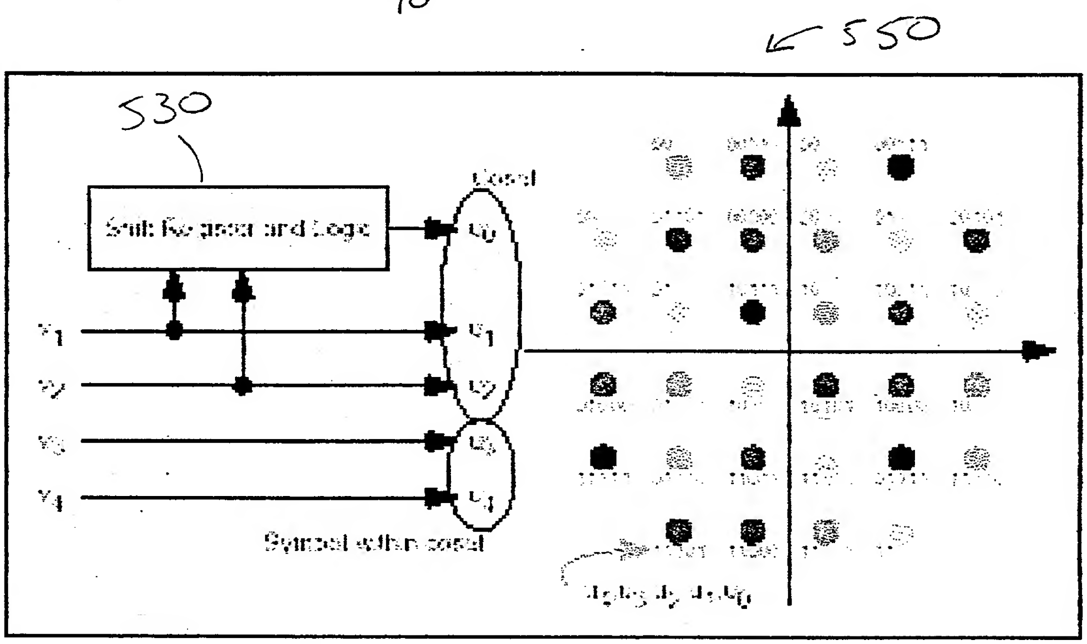


Figure ~~44~~ 41. ~~Generic Rate 4/5 TCM Encoder.~~

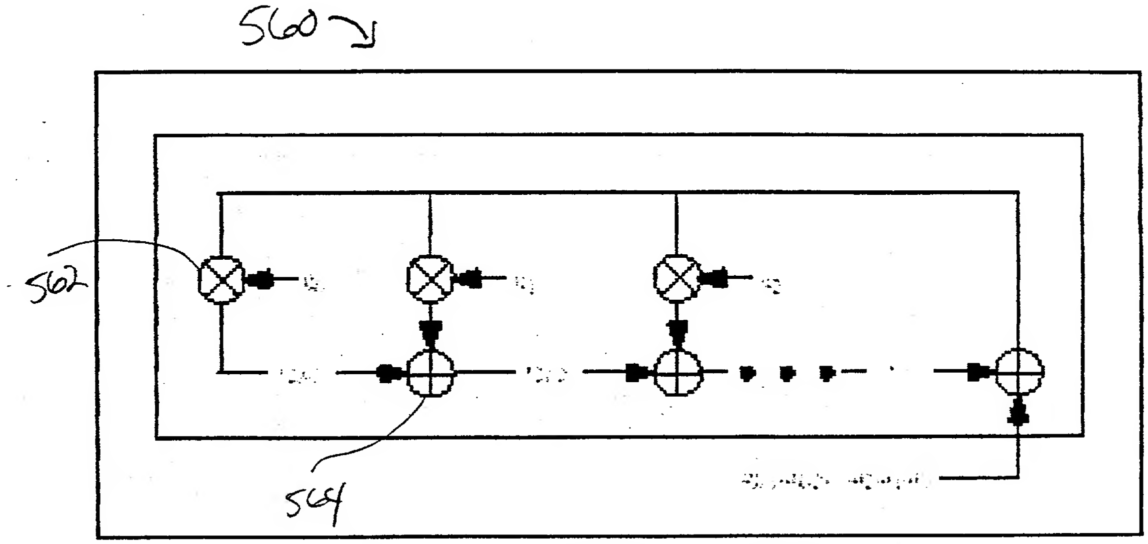


Figure ~~45.~~ ~~Generic Reed-Soloman Encoder.~~
42

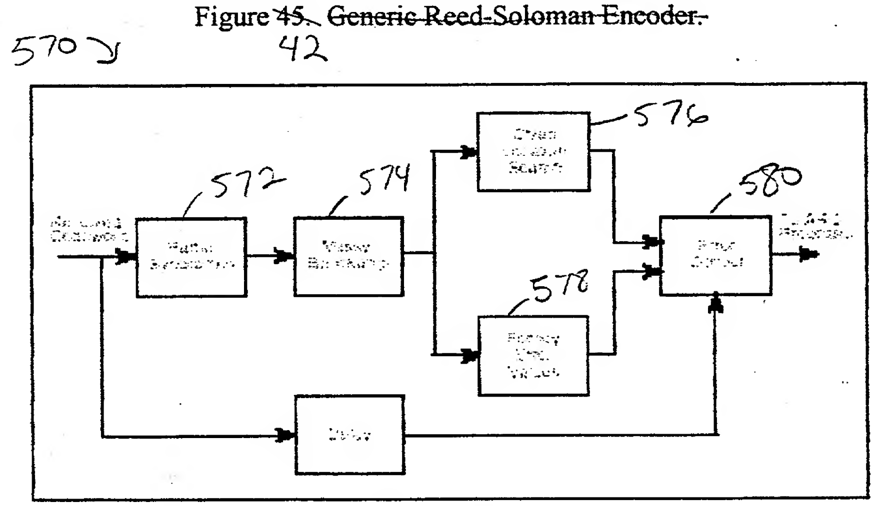


Figure ~~46.~~ ~~Generic Reed-Soloman Decoder.~~
43

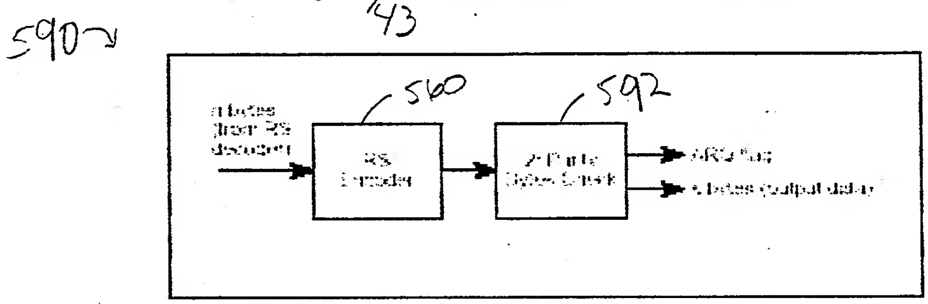


Figure ~~47.~~ ~~ARQ Processor.~~
44

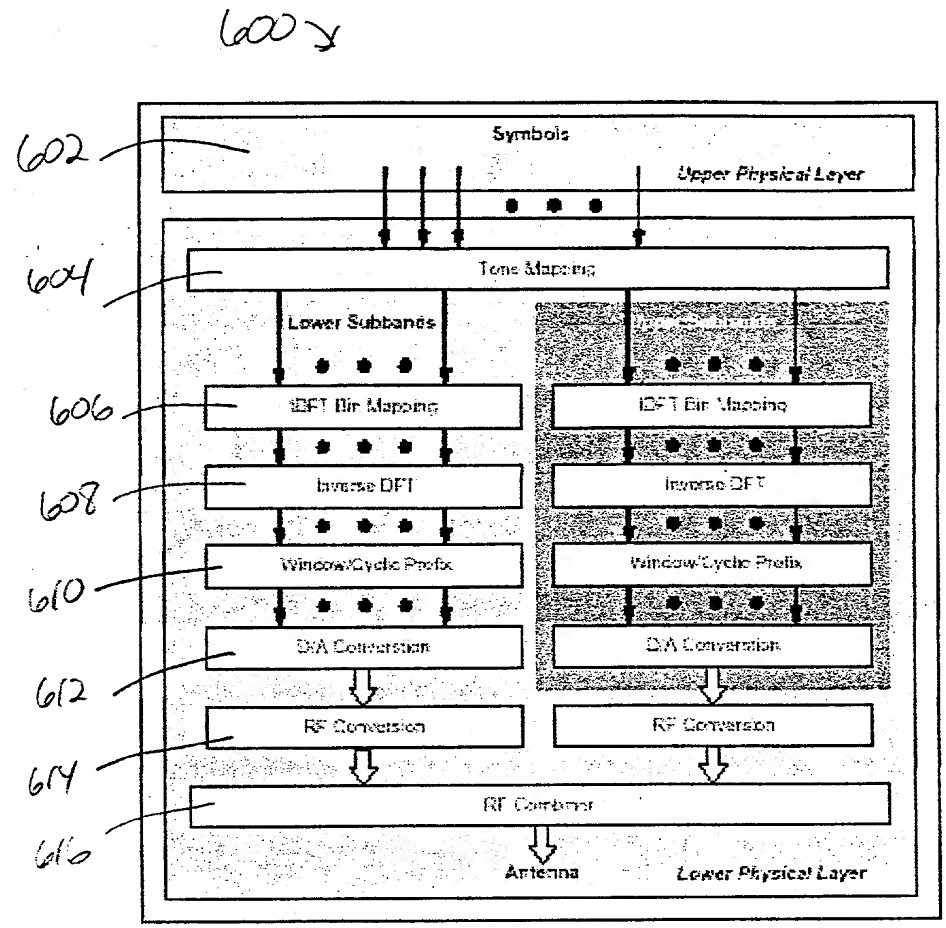


Figure ~~48.~~ 45 ~~Physical Layer Transmission Block Diagram.~~

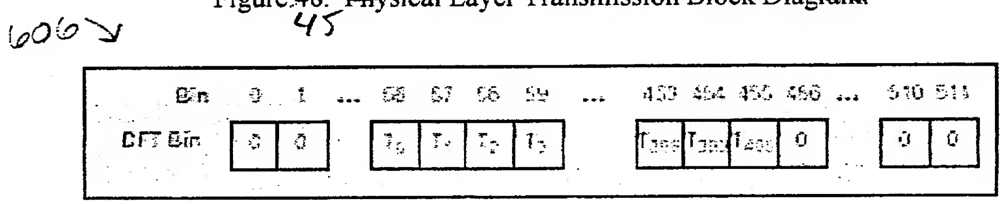


Figure ~~49.~~ 46 Mapping ~~of Tones into IDFT Bins.~~

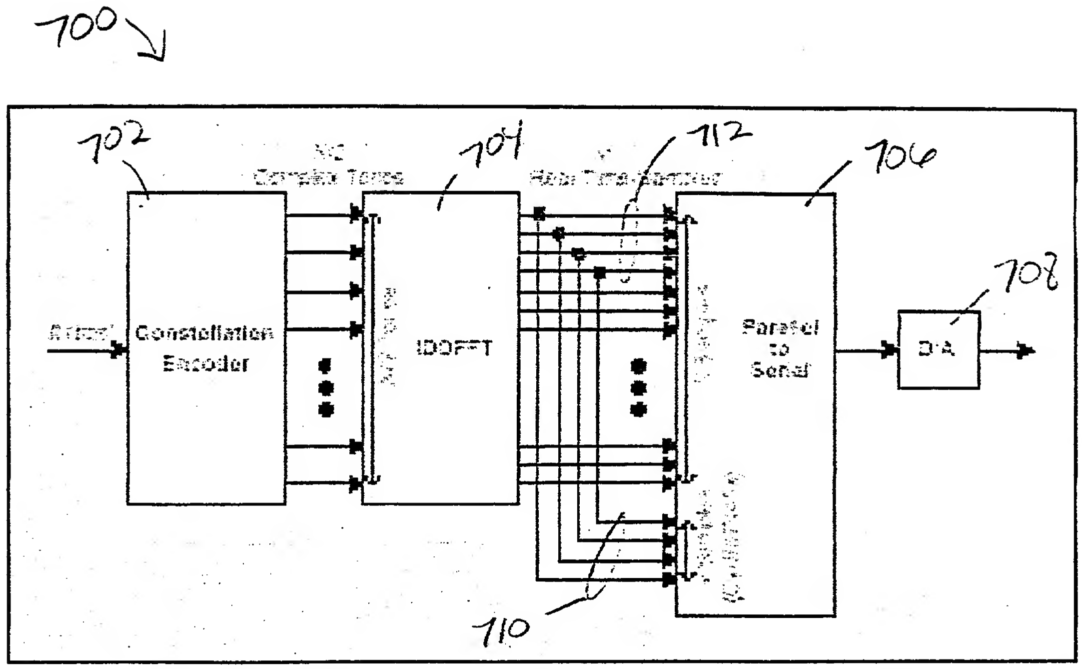


Figure ~~50~~. ~~Functional Synthesis Block Diagram with Cyclic Prefix.~~
47

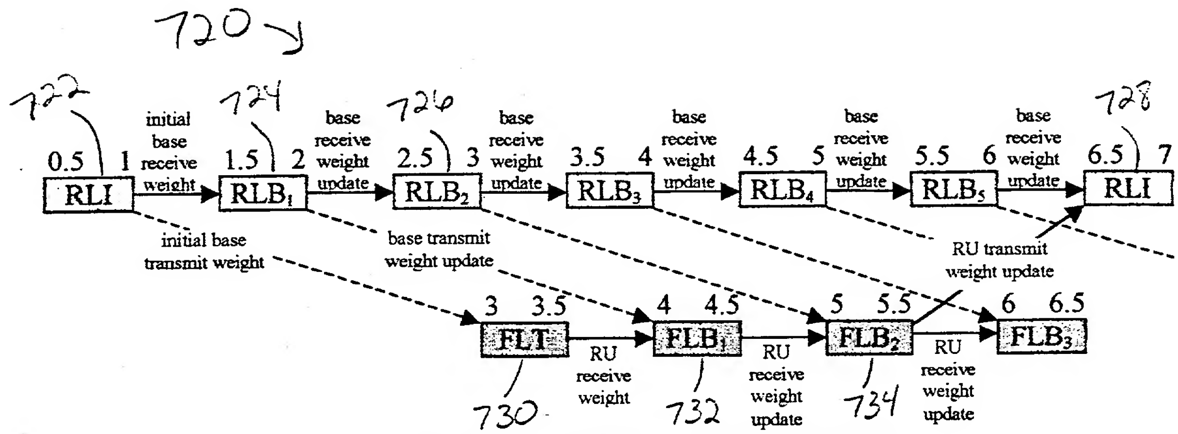


Figure ~~51.~~ 48 ~~Weight Vector Dependencies.~~

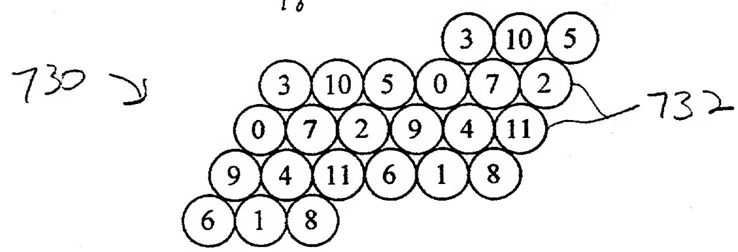


Figure ~~52.~~ 49 ~~Base offset codes for a repeat factor of 12, hexagonal layout.~~

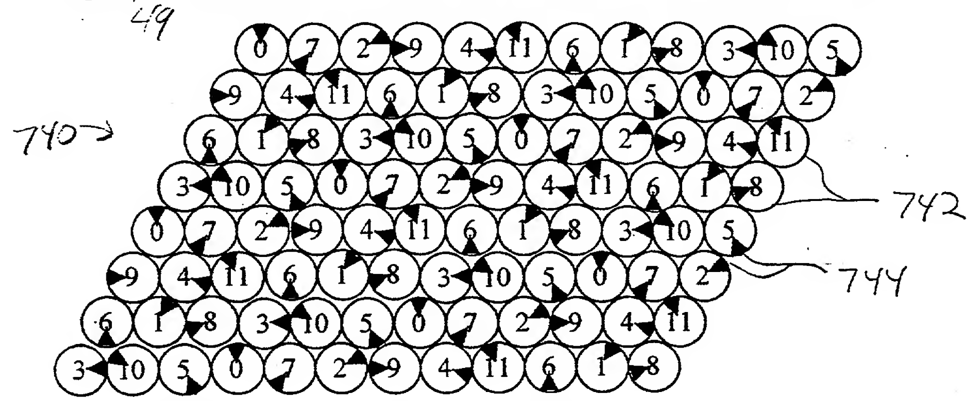


Figure ~~53.~~ 50 ~~Azimuths of a subset of FLI codes,~~ hexagonal layout ~~of base offset codes.~~

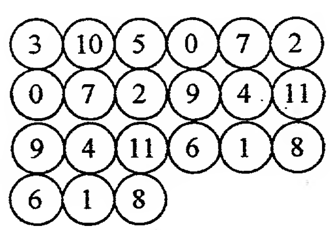


Figure ~~54.~~ 51 ~~Base offset codes for a repeat factor of 12, rectangular layout.~~

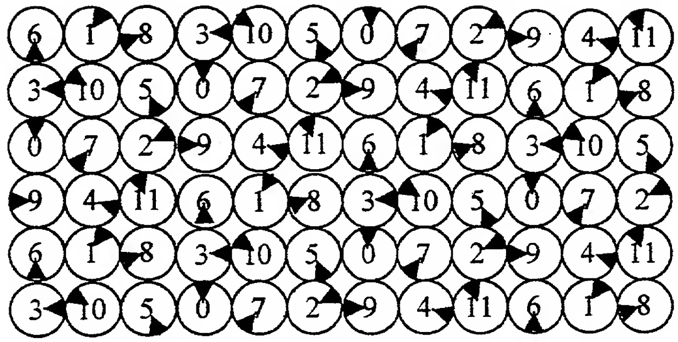


Figure ~~55~~. ~~Azimuths of a subset of access codes, rectangular layout of base offset codes.~~
52

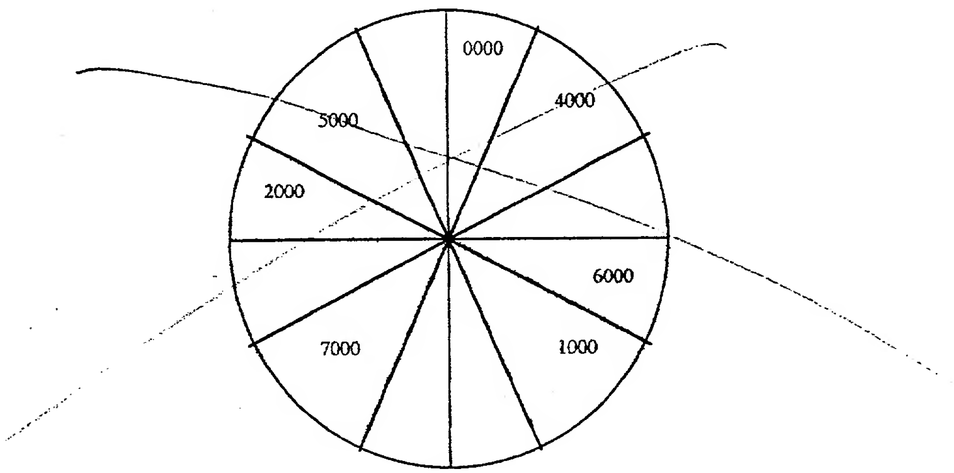


Figure ~~56.~~ Highlighted FLI access codes (expressed as octal digits) have high correlation with FLI access code 0000 and are at least 60° separated from 0000 in azimuth.

750

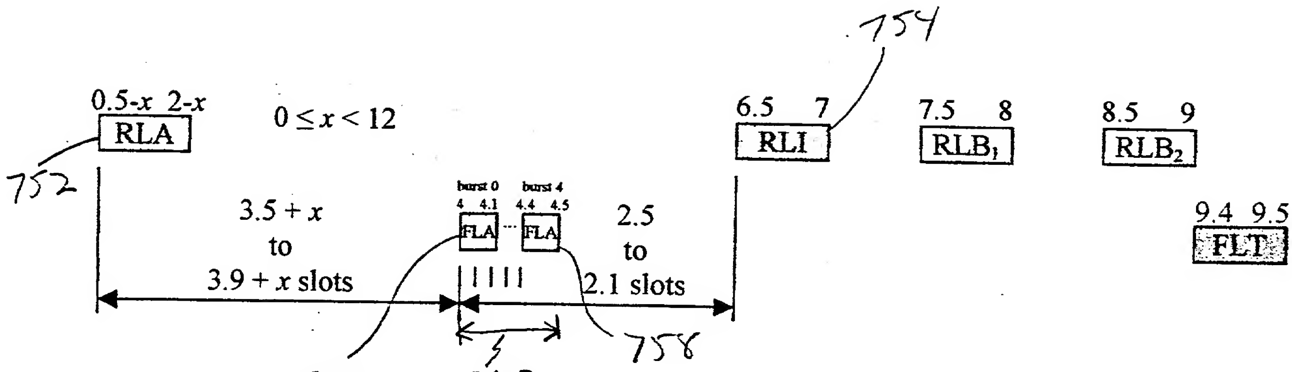


Figure ~~51~~ 53. ~~RLA, FLA, and RLI timing in slots.~~

Band	Bandwidth (MHz)	Subbands	Guard Bands (MHz)
WCS (A & B)	2 ×5	3	0.625
WCS (C/D)	2 ×5	2	1.250
MMDS	2 ×12	8	1.000

Figure ~~60~~ 54. ~~Subband Layouts.~~

Band of interest	Channel Bandwidth (MHz)	Number of Subbands	Active Bandwidth (MHz)	Guard Band on each side of active band. (MHz)
UHF, WCS, PCS	5	3	3.75	0.625
	10	7	8.75	0.625
	15	10	12.5	1.25
MMDS	3	2	2.5	0.25
	6	4	5	0.5
	12	8	10	1
3.5 GHz	3.5	2	2.5	0.5
	7	4	5	1
	14	8	10	2
3.65 GHz	25	16	20	2.5

Figure ~~61~~ 55. ~~Active bandwidth, Channelization, and Guard bands~~.

Tones per burst	16	16	16	16	16	16
Information bits per tone	4	4	3	3	2	2
Bits per burst	64	64	48	48	32	32
Bursts per slot	5	4	5	4	5	4
Bits per bearer slot	320	256	240	192	160	128
Bits per frame	1600	1280	1200	960	800	640
Partition rate (kbps)	80	64	60	48	40	32
Full rate (kbps)	1920	1536	1440	1152	960	768

Figure ~~62~~ 56. ~~Throughput.~~

i_1 \ i_0	$i_0=0$	$i_0=1$	$i_0=2$	$i_0=3$	$i_0=4$	$i_0=5$	$i_0=6$	$i_0=7$	…	$i_0=62$	$i_0=63$
$i_1=0$	X	a=0	a=64	a=128	a=192	…	a=3712	a=3776	a=3840	a=3904	a=3968
$i_1=1$	a=3969	X	a=1	a=65	a=129	…	a=3649	a=3713	a=3777	a=3841	a=3905
$i_1=2$	a=3906	a=3970	X	a=2	a=66	…	a=3586	a=3650	a=3714	a=3778	a=3842
$i_1=3$	a=3843	a=3907	a=3971	X	a=3	…	a=3523	a=3587	a=3651	a=3715	a=3779
$i_1=4$	a=3780	a=3844	a=3908	a=3972	X	…	a=3460	a=3524	a=3588	a=3652	a=3716
$i_1=5$	…	…	…	…	…	…	…	…	…	…	…
$i_1=6$	a=315	a=379	a=443	a=507	a=571	…	X	a=59	a=123	a=187	a=251
$i_1=7$	a=252	a=316	a=380	a=444	a=508	…	a=4028	X	a=60	a=124	a=188
…	a=189	a=253	a=317	a=381	a=445	…	a=3965	a=4029	X	a=61	a=125
$i_1=62$	a=126	a=190	a=254	a=318	a=382	…	a=3902	a=3966	a=4030	X	a=62
$i_1=63$	a=63	a=127	a=191	a=255	a=319	…	a=3839	a=3903	a=3967	a=4031	X

Figure 57. ~~RLI Access codes,~~ a as a function of the ~~in-phase column index i_1 and quadrature column index i_0.~~

replaced

i_1 \ i_0	i_0=0	i_0=1	i_0=2	i_0=3	i_0=4	i_0=5	i_0=6	i_0=7	…	i_0=62	i_0=63
i_1=0	X	k=0	k=64	k=128	k=192	…	k=3712	k=3776	k=3840	k=3904	k=3968
i_1=1	k=3969	X	k=1	k=65	k=129	…	k=3649	k=3713	k=3777	k=3841	k=3905
i_1=2	k=3906	k=3970	X	k=2	k=66	…	k=3586	k=3650	k=3714	k=3778	k=3842
i_1=3	k=3843	k=3907	k=3971	X	k=3	…	k=3523	k=3587	k=3651	k=3715	k=3779
i_1=4	k=3780	k=3844	k=3908	k=3972	X	…	k=3460	k=3524	k=3588	k=3652	k=3716
i_1=5	…	…	…	…	…	…	…	…	…	…	…
i_1=6	k=315	k=379	k=443	k=507	k=571	…	X	k=59	k=123	k=187	k=251
i_1=7	k=252	k=316	k=380	k=444	k=508	…	k=4028	X	k=60	k=124	k=188
…	k=189	k=253	k=317	k=381	k=445	…	k=3965	k=4029	X	k=61	k=125
i_1=62	k=126	k=190	k=254	k=318	k=382	…	k=3902	k=3966	k=4030	X	k=62
i_1=63	k=63	k=127	k=191	k=255	k=319	…	k=3839	k=3903	k=3967	k=4031	X

Figure ~~63.~~ ~~RLI Access codes, k as a function of the in-phase column index i_1 and quadrature column index i_0.~~

```
function fli = make_fli(codeword_descriptor)
% function fli = make_fli(codeword_descriptor)
% Synthesize a scaled 16 by 1 FLI codeword.
% 0 <= codeword_descriptor < 4096

% select the octal digits from the codeword descriptor
i0 = bitand(codeword_descriptor,7);
i1 = bitand(bitshift(codeword_descriptor,-3),7);
i2 = bitand(bitshift(codeword_descriptor,-6),7);
i3 = bitand(bitshift(codeword_descriptor,-9),7);
generatingVector = [i0, i1, i2, i3]; % generating vector

% the following kronecker basis function provides 4096 total codes
% and is based on an 8-star constellation
h = [ ...
      1.1923+0.2372j, 2.0960+0.4169j, 1.1923+0.2372j, 2.0960+0.4169j, ...
      1.1923+0.2372j, 2.0960+0.4169j, 1.1923+0.2372j, 2.0960+0.4169j; ...
      2.0960+0.4169j, 0.6754+1.0108j,-0.4169+2.0960j,-1.0108+0.6754j, ...
     -2.0960-0.4169j,-0.6754-1.0108j, 0.4169-2.0960j, 1.0108-0.6754j];

% make the kronecker codeword
fli = 1;
for jj=1:4
   fli = kron(h(:,generatingVector(jj)+1), fli); % matlab is one based
end

% quantize the codeword
fli = round(fli);
```

Figure ~~64.~~ ~~Matlab code to generate forward link codewords.~~

760

```
% fls_super_results_12.m
% Lower 12 bits are the base tones, upper 4 bits are the superframe tones.
% First index (row) is the base, second (column) is the superframe
codeword = [ ...
 23125 39509 27221 55893  6741 43605 47701 10837 51797 31317 59989 19029; ...
 40269 36173 64845 44365 56653 11597 48461 15693 27981 60749 52557 32077; ...
 47781 60069 27301 15013 10917 39589 51877  2725 35493 19109 43685 55973; ...
 13669 54629  5477 34149 62821 21861  9573 38245 42341 46437 30053 50533; ...
 27309 10925 55981 43693 47789 51885  6829 35501 15021 19117 39597 23213; ...
 21813 38197 34101  5429 42293 54581  9525 62773 46389 17717 50485 58677; ...
 27477 56149 11093 43861 19285 39765  6997 23381 52053 35669 60245 47957; ...
 42389 17813 46485 50581 21909  1429  9621 62869 30101 54677 26005 58773; ...
 42709 38613 46805 14037 18133 50901  5845 22229 54997 59093 34517 30421; ...
 38217 46409 25929 42313  5449  9545 50505 13641 54601 17737 21833 30025; ...
  4693 12885 21077 16981 53845 41557 49749 62037 45653 29269 25173 37461; ...
 59049 34473  5801  9897 54953 13993 26281 18089 38569 42665 46761 50857; ...
];

%  5A55  9A55  6A55  DA55  1A55  AA55  BA55  2A55  CA55  7A55  EA55  4A55
%  9D4D  8D4D  FD4D  AD4D  DD4D  2D4D  BD4D  3D4D  6D4D  ED4D  CD4D  7D4D
%  BAA5  EAA5  6AA5  3AA5  2AA5  9AA5  CAA5  0AA5  8AA5  4AA5  AAA5  DAA5
%  3565  D565  1565  8565  F565  5565  2565  9565  A565  B565  7565  C565
%  6AAD  2AAD  DAAD  AAAD  BAAD  CAAD  1AAD  8AAD  3AAD  4AAD  9AAD  5AAD
%  5535  9535  8535  1535  A535  D535  2535  F535  B535  4535  C535  E535
%  6B55  DB55  2B55  AB55  4B55  9B55  1B55  5B55  CB55  8B55  EB55  BB55
%  A595  4595  B595  C595  5595  0595  2595  F595  7595  D595  6595  E595
%  A6D5  96D5  B6D5  36D5  46D5  C6D5  16D5  56D5  D6D5  E6D5  86D5  76D5
%  9549  B549  6549  A549  1549  2549  C549  3549  D549  4549  5549  7549
%  1255  3255  5255  4255  D255  A255  C255  F255  B255  7255  6255  9255
%  E6A9  86A9  16A9  26A9  D6A9  36A9  66A9  46A9  96A9  A6A9  B6A9  C6A9
%
Nb = 12; % Number of tones in base
Ns =  4; % Number of tones in superframe sequence
Nt = 16; % Total number of tones
```

762

764

Figure ~~65.~~ ~~Matlab code for FLS codeword descriptors.~~
59

```
function fls = make_fls(base, superframe)
% function fls = make_fls(base, superframe)
% Synthesize a scaled 16 by 1 FLS codeword.
% base is the base offset code and varies from 0 to 11
% superframe is the slot sequence number and varies from 0 to 11

fls_super_results_12 % read in the codeword descriptor array

t = zeros(Nt,1);
for jj=1:Nt
   t(jj) = 2^(jj-1); % form a vector of walking ones
end

cw = codeword(base+1, superframe+1); % select codeword descriptor
bv = (bitand(cw,t) ~= 0) * 2 - 1; % make BPSK vector
fls = (15 + 15j) * bv; % scale the BPSK vector
```

Figure ~~66.~~ ~~Matlab code to synthesize FLS codewords~~
60

770

772a

774

772b

Partition			0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24
Frame	Slot	Burst																									
0	0	0	0						0						0						0						
		3	1						1						1						1						
0	1	0		1						1						1						1					
		3		0						0						0						0					
0	2	0			2						2						2						2				
		3			3						3						3						3				
0	3	0				3						3						3						3			
		3				2						2						2						2			
0	4	0					4						4						4						4		
		3					5						5						5						5		
0	5	0						5						5						5						5	
		3						4						4						4						4	
1	6	0	6						6						6						6						
		3	7						7						7						7						
1	7	0		7						7						7						7					
		3		6						6						6						6					
1	8	0			8						8						8						8				
		3			9						9						9						9				
1	9	0				9						9						9						9			
		3				8						8						8						8			
1	10	0					10						10						10						10		
		3					11						11						11						11		
1	11	0						11						11						11						11	
		3						10						10						10						10	
2	12	0	0						0						0						0						
		3	1						1						1						1						
2	13	0		1						1						1						1					
		3		0						0						0						0					
2	14	0			2						2						2						2				
		3			3						3						3						3				
2	15	0				3						3						3						3			
		3				2						2						2						2			
2	16	0					4						4						4						4		
		3					5						5						5						5		
2	17	0						5						5						5						5	
		3						4						4						4						4	
3	18	0	6						6						6						6						
		3	7						7						7						7						
3	19	0		7						7						7						7					
		3		6						6						6						6					
3	20	0			8						8						8						8				
		3			9						9						9						9				
3	21	0				9						9						9						9			
		3				8						8						8						8			
3	22	0					10						10						10						10		
		3					11						11						11						11		
3	23	0						11						11						11						11	
		3						10						10						10						10	

Figure ~~67~~ 61. ~~FLS codeword number sequence for a spreading factor of 2.~~

Partition			0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24
Frame	Slot	Burst																									
0	0	0	0			1			0			1			0			1			0			1			
		3	1			0			1			0			1			0			1			0			
0	1	0																									
		3																									
0	2	0		2			3			2			3			2			3			2			3		
		3		3			2			3			2			3			2			3			2		
0	3	0																									
		3																									
0	4	0			4			5			4			5			4			5			4			5	
		3			5			4			5			4			5			4			5			4	
0	5	0																									
		3																									
1	6	0	7			6			7			6			7			6			7			6			
		3	6			7			6			7			6			7			6			7			
1	7	0																									
		3																									
1	8	0		9			8			9			8			9			8			9			8		
		3		8			9			8			9			8			9			8			9		
1	9	0																									
		3																									
1	10	0			11			10			11			10			11			10			11			10	
		3			10			11			10			11			10			11			10			11	
1	11	0																									
		3																									
2	12	0	0			1			0			1			0			1			0			1			
		3	1			0			1			0			1			0			1			0			
2	13	0																									
		3																									
2	14	0		2			3			2			3			2			3			2			3		
		3		3			2			3			2			3			2			3			2		
2	15	0																									
		3																									
2	16	0			4			5			4			5			4			5			4			5	
		3			5			4			5			4			5			4			5			4	
2	17	0																									
		3																									
3	18	0	7			6			7			6			7			6			7			6			
		3	6			7			6			7			6			7			6			7			
3	19	0																									
		3																									
3	20	0		9			8			9			8			9			8			9			8		
		3		8			9			8			9			8			9			8			9		
3	21	0																									
		3																									
3	22	0			11			10			11			10			11			10			11			10	
		3			10			11			10			11			10			11			10			11	
3	23	0																									
		3																									

Figure ~~68~~. ~~FLS codeword number sequence for a spreading factor of 4.~~
62

Partition			0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24
Frame	Slot	Burst																									
0	0	0 3	a a						α α						Υ Υ						Ψ Ψ						
0	1	0 3		b b						β β						Χ Χ						Ξ Ξ					
0	2	0 3			c c						χ χ						W W						Ω Ω				
0	3	0 3				d d						δ δ						V V						ς ς			
0	4	0 3					e e						ε ε						U U						∴ ∴		
0	5	0 3						f f						ϕ ϕ						T T						| |	
1	6	0 3	S S						γ γ						g g						Σ Σ						
1	7	0 3		R R						η η						h h						[[					
1	8	0 3			Q Q						ι ι						i i						Θ Θ				
1	9	0 3				P P						φ φ						j j						Π Π			
1	10	0 3					O O						κ κ						k k						} }		
1	11	0 3						N N						λ λ						l l						]]	
2	12	0 3	L L						Λ Λ						m m						μ μ						
2	13	0 3		K K						))						n n						ν ν					
2	14	0 3			J J						ϑ ϑ						o o						~ ~				
2	15	0 3				I I						* *						p p						π π			
2	16	0 3					H H						((						q q						θ θ		
2	17	0 3						G G						Γ Γ						r r						ρ ρ	
3	18	0 3	s s						Φ Φ						F F						σ σ						
3	19	0 3		t t						& &						E E						τ τ					
3	20	0 3			u u						Δ Δ						D D						υ υ				
3	21	0 3				v v						X X						C C						ϖ ϖ			
3	22	0 3					w w						⊥ ⊥						B B						ω ω		
3	23	0 3						x x						% %						A A						ξ ξ	

Figure ~~69~~. ~~Base transmit weight patterns for FLS bursts for a spreading factor of 2.~~
63

Burst ->	0	1	2	3	4
Time slot counter modulo 6	Partition in which the RU is directed to send an RLI				
0	20	2	8	14	20
1	3	9	15	21	3
2	10	16	22	4	10
3	17	23	5	11	17
4	0	6	12	18	0
5	13	19	1	7	13

Figure ~~70~~ 64. ~~Mapping of FLAs to RLI partition numbers for 2-way spreading.~~

Burst ->	0	1	2	3	4
Time slot counter modulo 6	Partition in which the RU is directed to send an RLI				
0	8	11	2	5	8
1	3	6	9	0	3
2	10	1	4	7	10
3	5	8	11	2	5
4	0	3	6	9	0
5	1	4	7	10	1

Figure ~~71~~ 65. ~~Mapping of FLAs to RLI partition numbers for 4-way spreading.~~

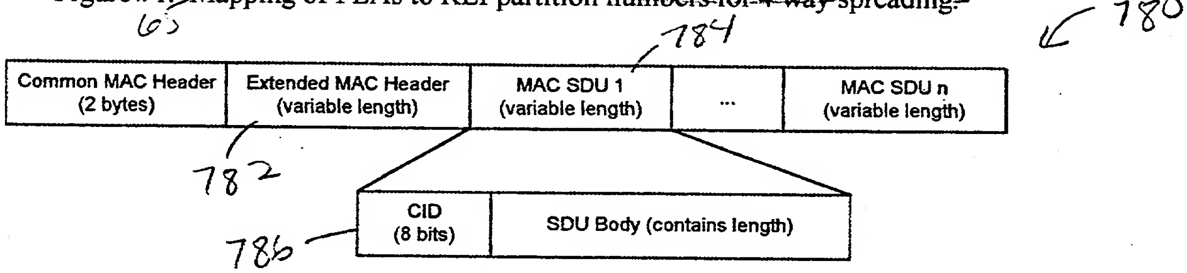


Figure ~~72~~ 66. ~~MAC frame structure~~

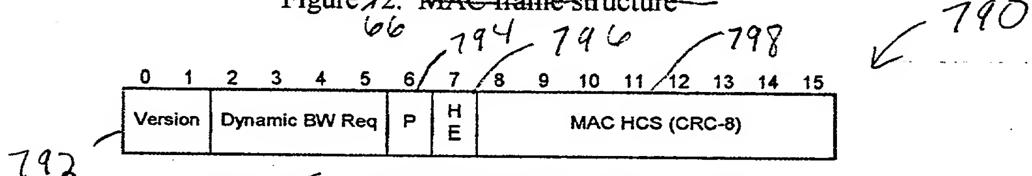


Figure ~~73~~ 67. Reverse link common MAC header

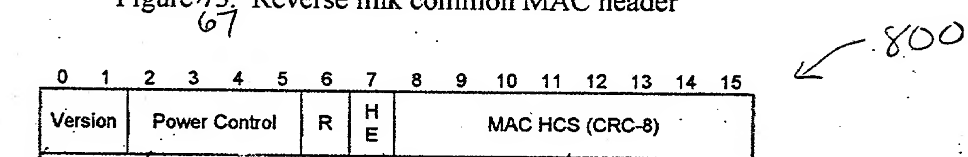


Figure ~~74~~ 68. Forward link MAC common header

808 804 806 810

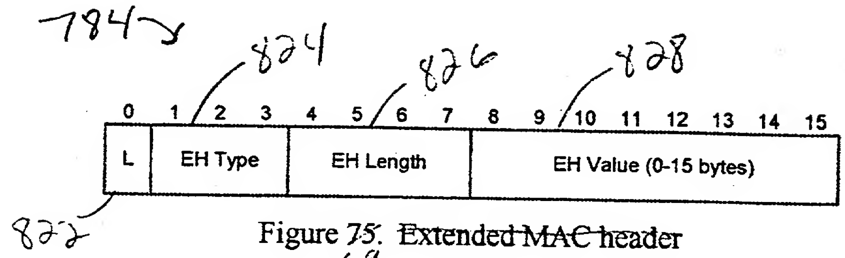


Figure ~~75~~. ~~Extended MAC header~~
69

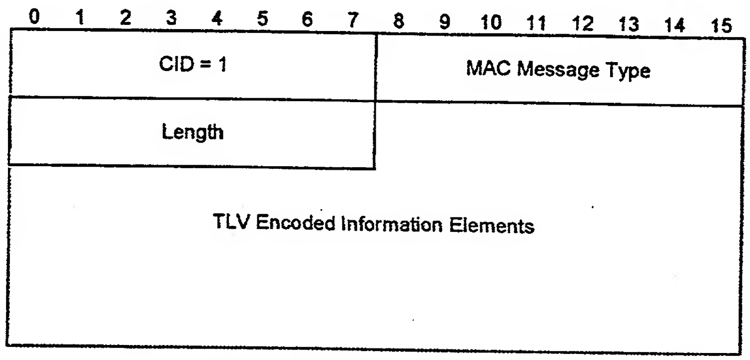
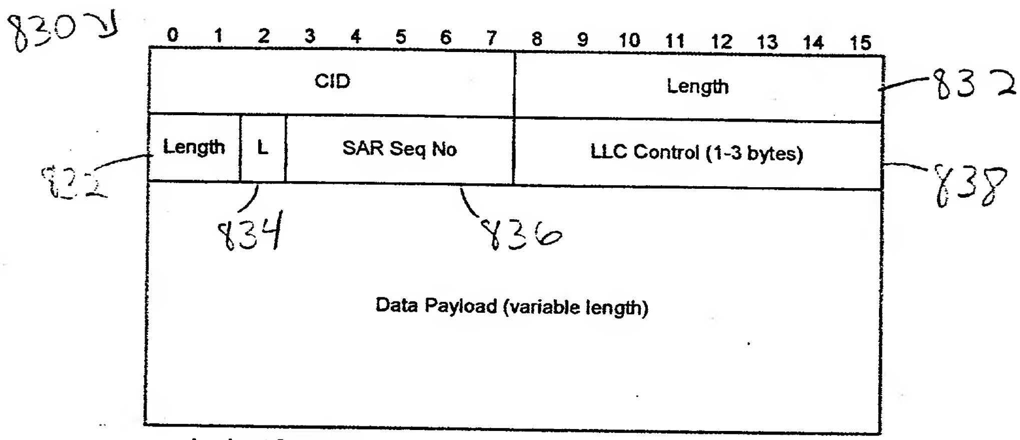


Figure ~~76~~. ~~MAC message format~~
70



- **L = Last Segment**

Figure ~~77~~. ~~Data SDU format~~
71

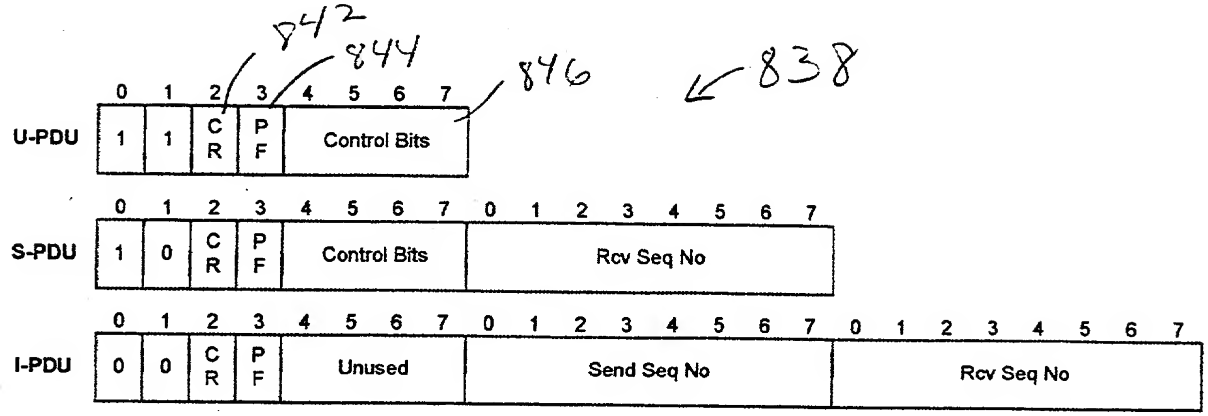


Figure ~~78~~ 72. LLC control field format

Modulation Order	4 Bits/Sym		3 Bits/Sym		2 Bits/Sym	
Link Direction	Forward	Reverse	Forward	Reverse	Forward	Reverse
Bits/Symbol	4	4	3	3	2	2
Symbols/Burst	16	16	16	16	16	16
Bursts/Slot	5	4	5	4	5	4
Bits/Slot	320	256	240	192	160	128
Bytes/Slot	40	32	30	24	20	16
Slots/Frame	5	5	5	5	5	5
Bits/Frame	1600	1280	1200	960	800	640
Bytes/Frame	200	160	150	120	100	80
Viterbi Tail Byte(*)	1	1	1	1	1	1
RS Check Bytes	28	28	18	18	10	10
Common MAC Header	2	2	2	2	2	2
MAC SDU Length	169	129	129	99	87	67
Data SDU Header	6	6	6	6	6	6
Data Payload	163	123	123	93	81	61
Data Rate/Partition, kbps	65.2	49.2	49.2	37.2	32.4	24.4
Partitions/Subband	24	24	24	24	24	24
Data Rate/Subband, kbps	1564.8	1180.8	1180.8	892.8	777.6	585.6
Subband Data Rate/T1	1.02	0.77	0.77	0.58	0.51	0.38

Figure ~~79~~. ~~Frame sizes for 20 ms frames~~
73

<table>
<tr><th>0</th><th>1</th><th>2</th><th>3</th><th>4</th><th>5</th><th>6</th><th>7</th><th>8</th><th>9</th><th>10</th><th>11</th><th>12</th><th>13</th><th>14</th><th>15</th></tr>
<tr><td colspan="8">CID</td><td colspan="4">Length (coded)</td><td>P T</td><td>L</td><td colspan="2">SAR Seq No</td></tr>
<tr><td colspan="16">IP Identification</td></tr>
<tr><td colspan="16">IP HCS</td></tr>
<tr><td colspan="16">UDP HCS</td></tr>
<tr><td colspan="16">RTP Seq No</td></tr>
<tr><td colspan="16">RTP Time Stamp</td></tr>
<tr><td colspan="16">VoIP Payload (variable length)</td></tr>
<tr><td colspan="16">Ethernet FCS</td></tr>
</table>

Figure ~~80.~~ 74 ~~20ms VoIP frame~~

<table>
<tr><th>0</th><th>1</th><th>2</th><th>3</th><th>4</th><th>5</th><th>6</th><th>7</th><th>8</th><th>9</th><th>10</th><th>11</th><th>12</th><th>13</th><th>14</th><th>15</th></tr>
<tr><td colspan="8">CID</td><td colspan="4">Length (coded)</td><td>P T</td><td>L</td><td colspan="2">SAR Seq No</td></tr>
<tr><td colspan="8">Compressed RTP Header</td><td colspan="8">UDP HCS</td></tr>
<tr><td colspan="8">UDP HCS</td><td colspan="8">VoIP HCS (CRC-8)</td></tr>
<tr><td colspan="16">VoIP Payload (variable length)</td></tr>
</table>

Figure ~~81.~~ 75 ~~10ms VoIP frame~~

Modulation Order	4 Bits/Sym		3 Bits/Sym		2 Bits/Sym		
Link Direction	Forward	Reverse	Forward	Reverse	Forward	Reverse	
Entry Slot	–	–	–	–	–	–	10 ms
Bearer Slot 1	40	32	30	24	20	16	
Bearer Slot 2	40	32	30	24	20	16	
Common MAC Header	2	2	2	2	2	2	
MAC SDU Length	78	62	58	46	38	30	
Bearer Slot 3	40	32	30	24	20	16	10 ms
Bearer Slot 4	40	32	30	24	20	16	
Bearer Slot 5(*)	39	31	29	23	19	15	
Common MAC Header	2	2	2	2	2	2	
MAC SDU Length	117	93	87	69	57	45	

(*) Viterbi tail byte occurs in the 5th bearer slot.

Figure ~~82~~. ~~10ms frame sizes.~~
76

Frame Duration	20ms			10 ms		
Vocoder	G.711	G.726	G.729	G.711	G.726	G.729
Bit Rate, kbps	64.0	32.0	8.0	64.0	32.0	8.0
Voice Bytes	160	80	20	80	40	10
VoIP Overhead(*)	16	16	16	3	3	3
VoIP Payload Size	176	96	36	83	43	13
Voice SDU Header	2	2	2	3	3	3
4 Bits/Sym						
SDU Size Limit (RL)	129	129	129	62	62	62
No. Partitions	2	1	1/3	2	1	1/3
SDU Size	90x2	98	38	45+44	46	16
3 Bits/Sym						
SDU Size Limit (RL)	99	99	99	46	46	46
No. Partitions	2	1	1/2	2	1	1/2
SDU Size	90x2	98	38	45+44	46	16
2 Bits/Sym						
SDU Size Limit (RL)	67	67	67	30	30	30
No. Partitions	3	2	1	4	2	1
SDU Size	61x2+60	50x2	38	24x3+23	25+24	16

(*) Include RTP, UDP, IP, PPPoE, and Ethernet

Figure ~~83~~. ~~VoIP payload sizes~~
77

850

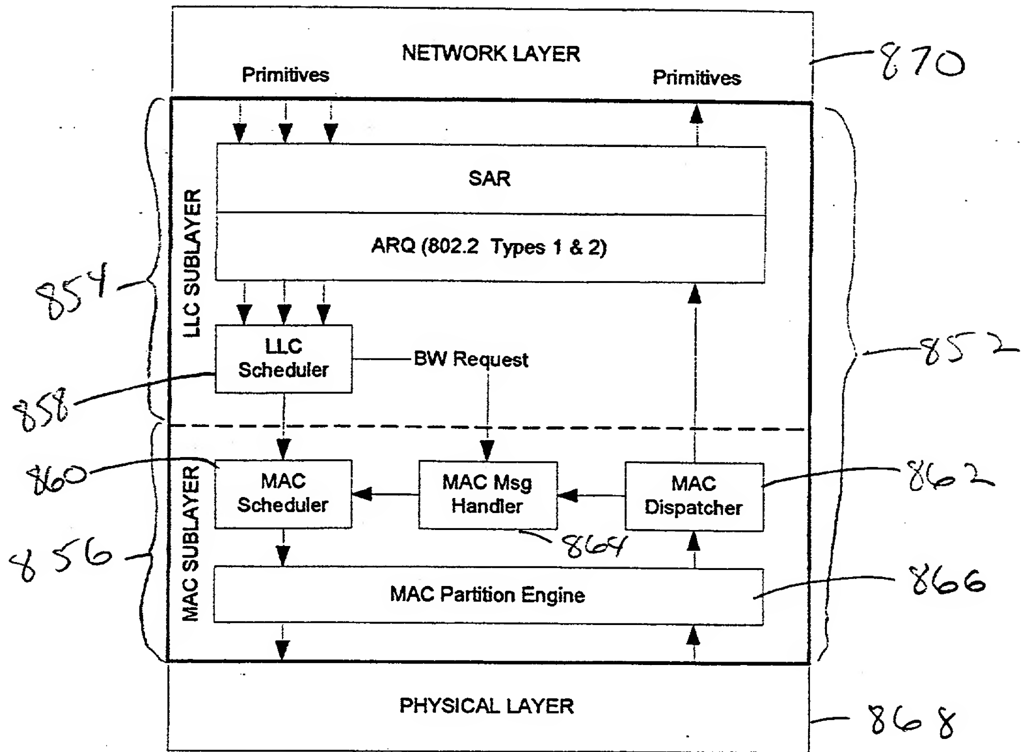


Figure ~~84~~ 78. ~~DLL layer components~~

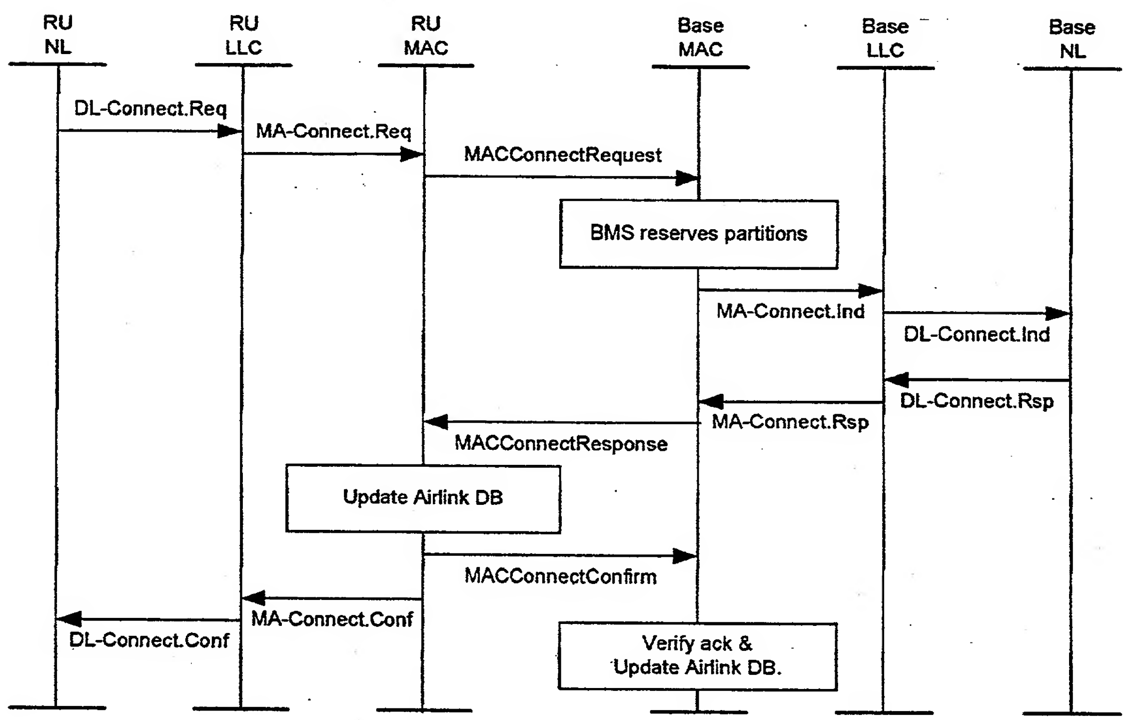


Figure ~~85~~ 79. ~~Voice setup illustration~~

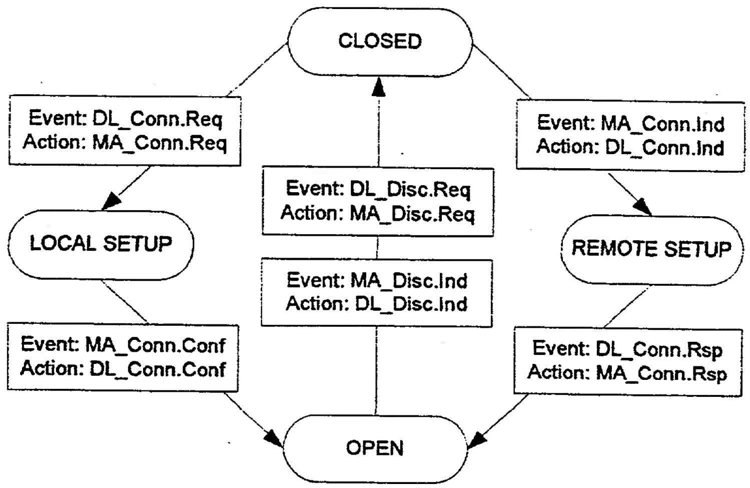


Figure ~~86~~. LLC state diagram for voice
80

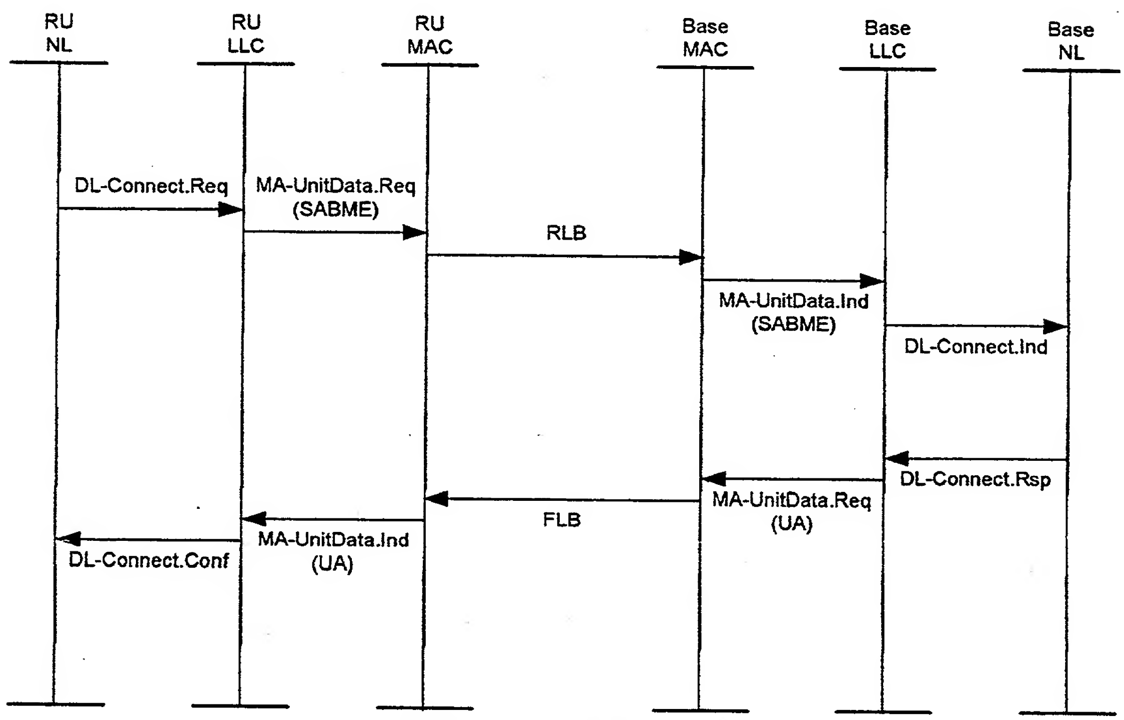
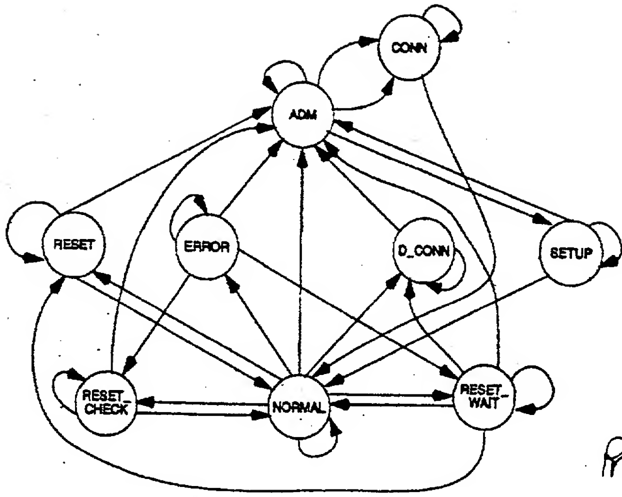


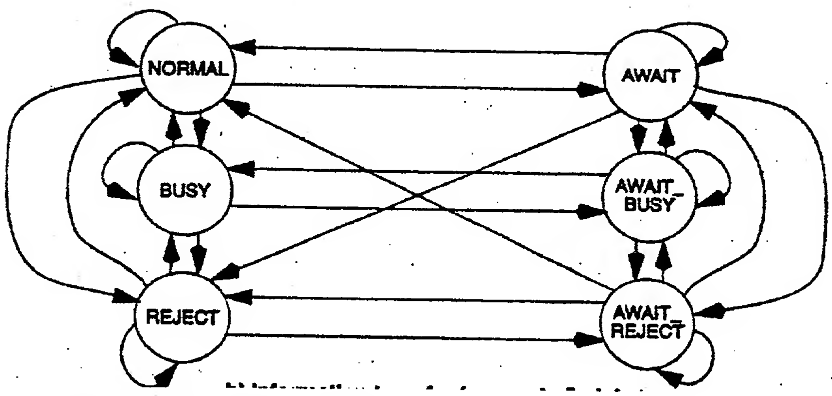
Figure ~~87~~. ~~Illustration of data primitives~~
81



Prior Art

82

Figure ~~88~~. LLC ~~state diagram link setup, teardown and recovery phase, prior art.~~



Prior Art

Figure ~~89~~. ~~LLC state diagram information transfer phase~~

83

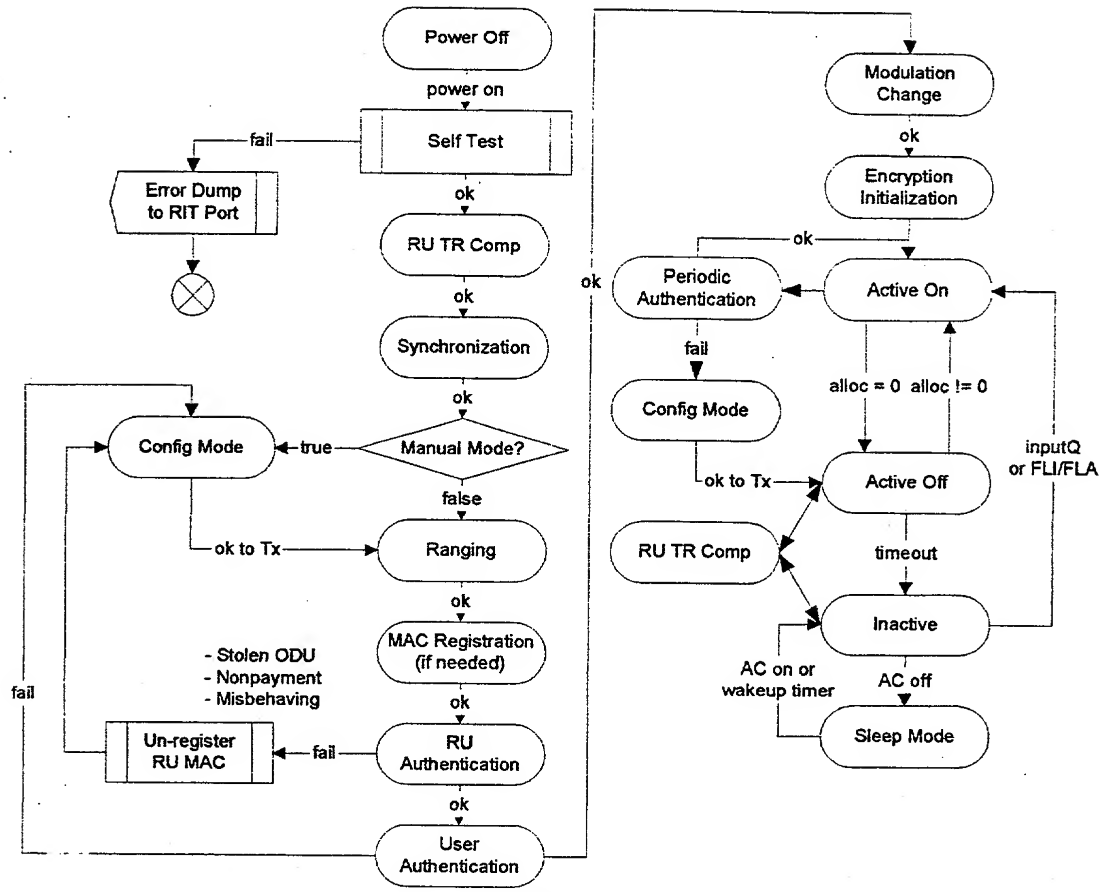


Figure ~~90~~ 84. ~~RU ODU state diagram.~~

882

880

884

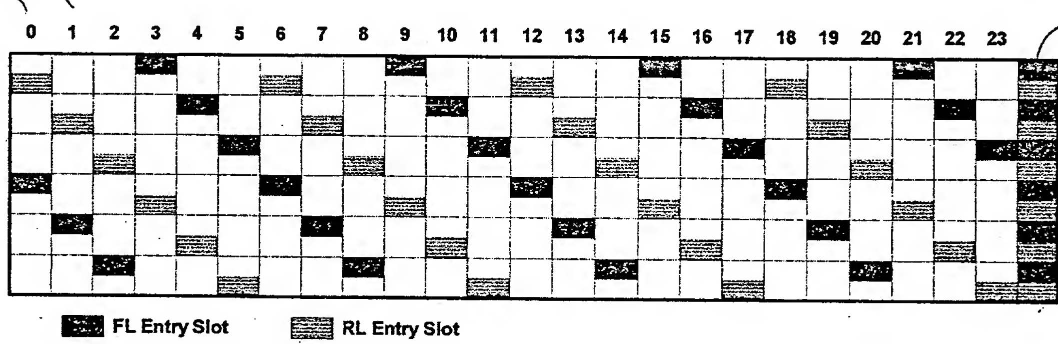


Figure ~~91~~ 85. ~~Airlink frame structure~~

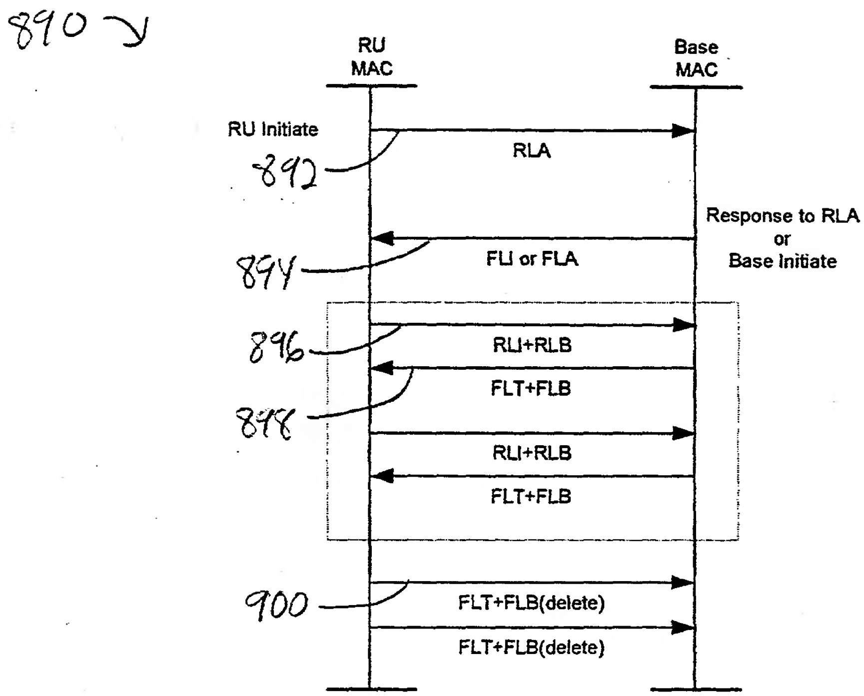


Figure ~~92~~. ~~Connection initiation and data transfer~~ 86

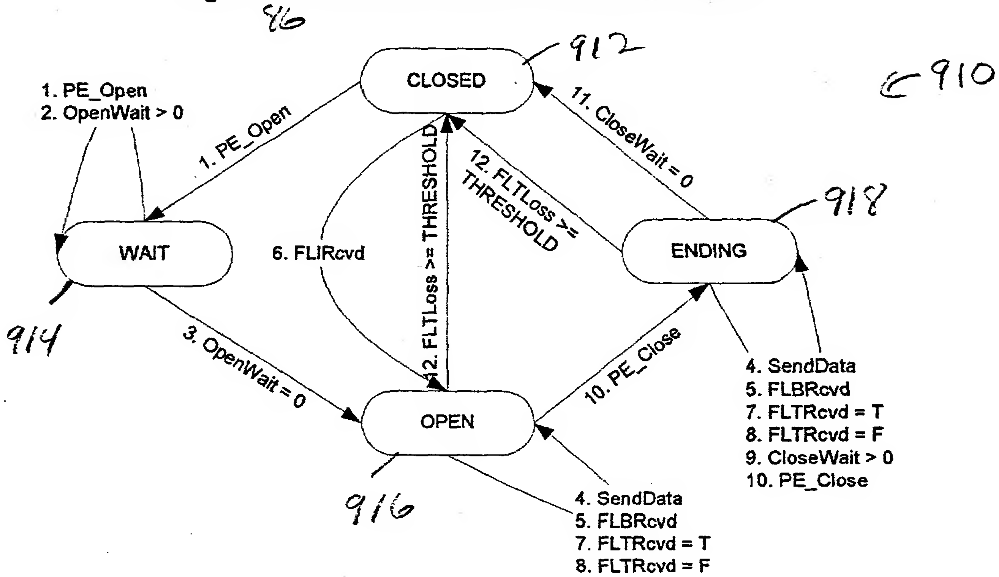


Figure ~~93~~. ~~RU PE frame-driven component~~ 87

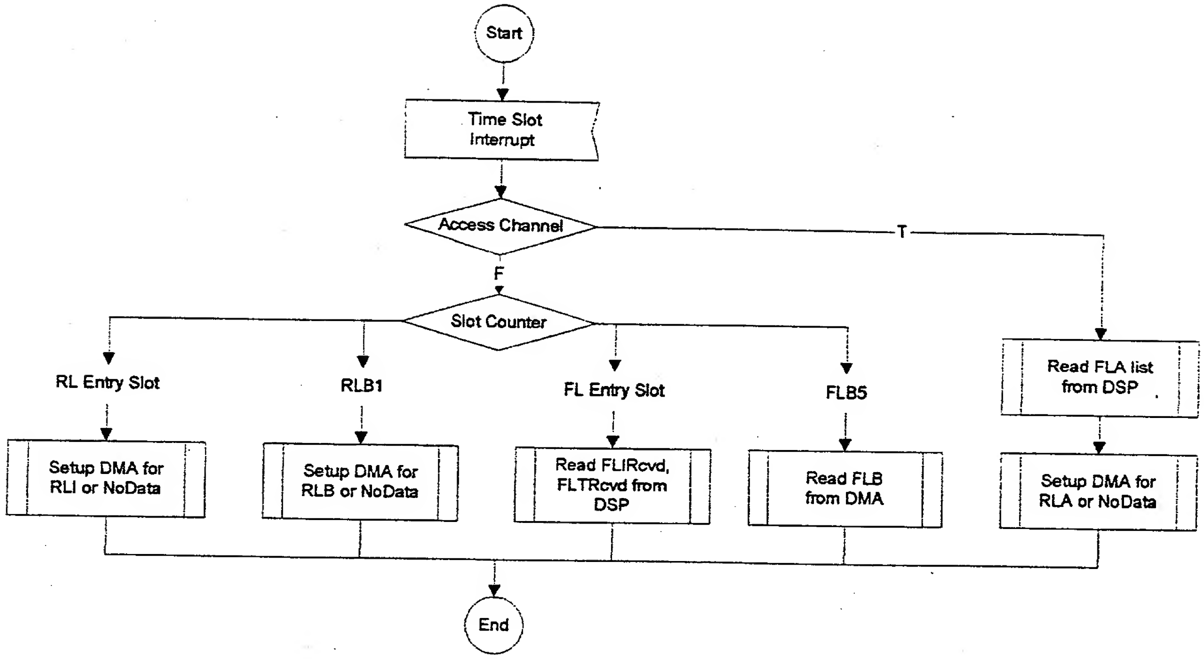


Figure ~~94.~~ ~~RU PE slot-driven component~~
88

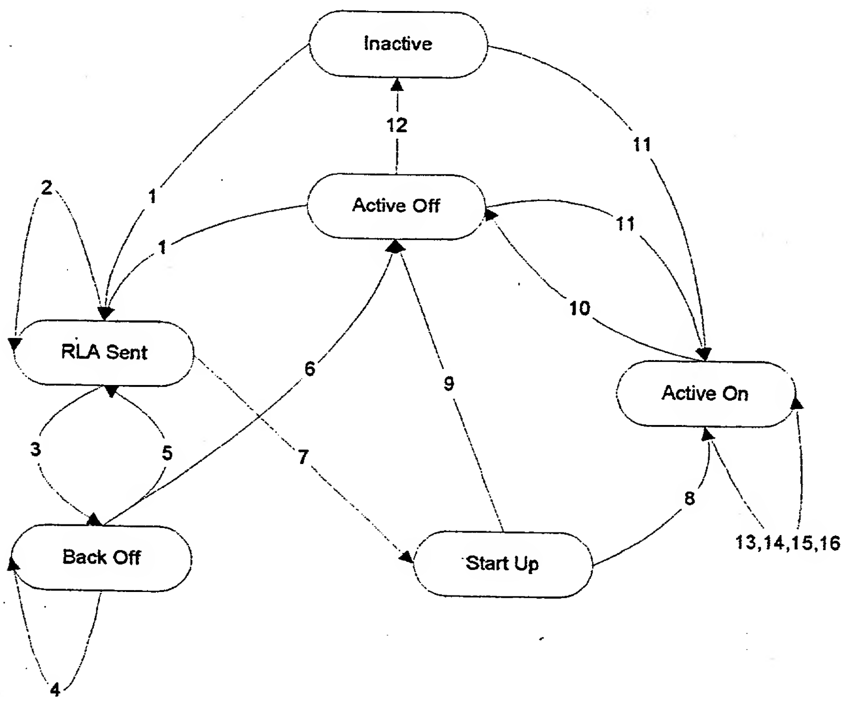


Figure ~~95.~~ ~~RU MAC scheduler state diagram.~~
89

1. Event = MA_UnitData.Request
SendRLA; Set ReplyCounter; RLAMiss=0;

2. Event = ReplyTimer > 0
ReplyCounter–;

3. Event = ReplyCounter = 0
RLAMiss++; BORetry–;
BOCounter=Ran(MIN,MIN+2^RLAMiss*Win);

4. Event = BOCounter>0
BOCounter–;

5. Event = BOCounter=0 & BORetry>0
RLAMiss=0; SendRLA; Set ReplyCounter;

6. Event = BOCounter=0 & BORetry=0
Issue access failure signal; Reset BORetry;

7. Event = FLIRcvd or FLARcvd
Start PE to add partition; wait for partition open

8. Event = PE Success

9. Event = PE Fail
Issue access failure signal (?)

10. Event = Delete last partition
Start PE to delete partition;

11. Event = FLIRcvd or FLARcvd
Start PE to add partition

12. Event = ActiveOffTimeout
Reinitialize encryption/scrambling engines (call PE)

13.Event = MA_UnitData.Request
PE_SendData

14. Event = FLBRcvd
PE_UnitData.Indication

Figure ~~96~~. ~~Events and actions of RU MAC scheduler~~
90

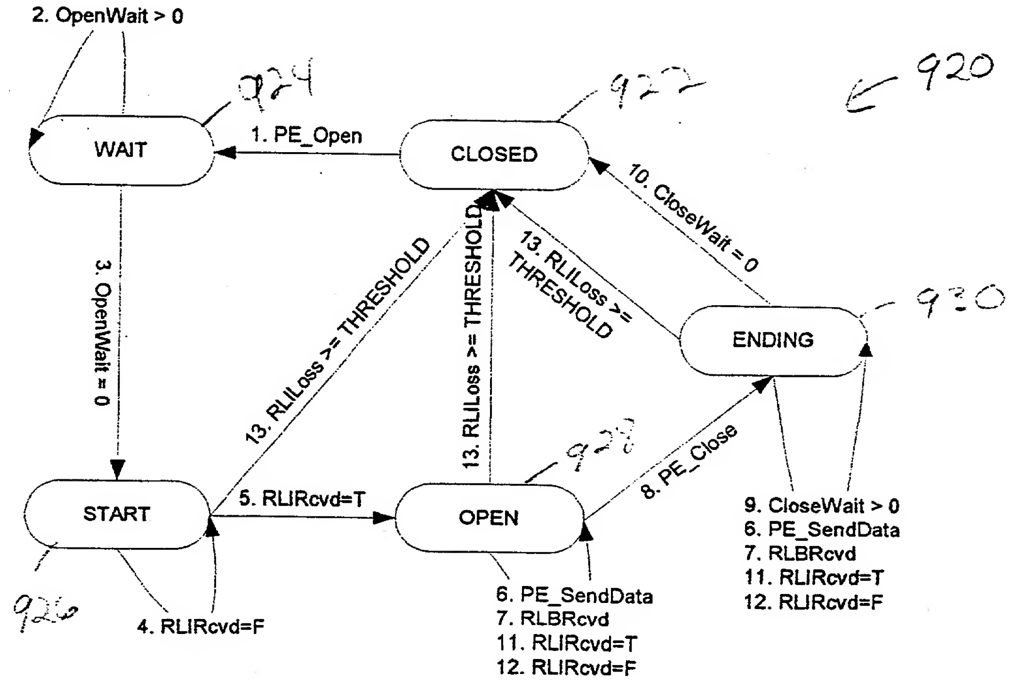


Figure ~~97~~. ~~Base partition engine – frame driven component~~
91

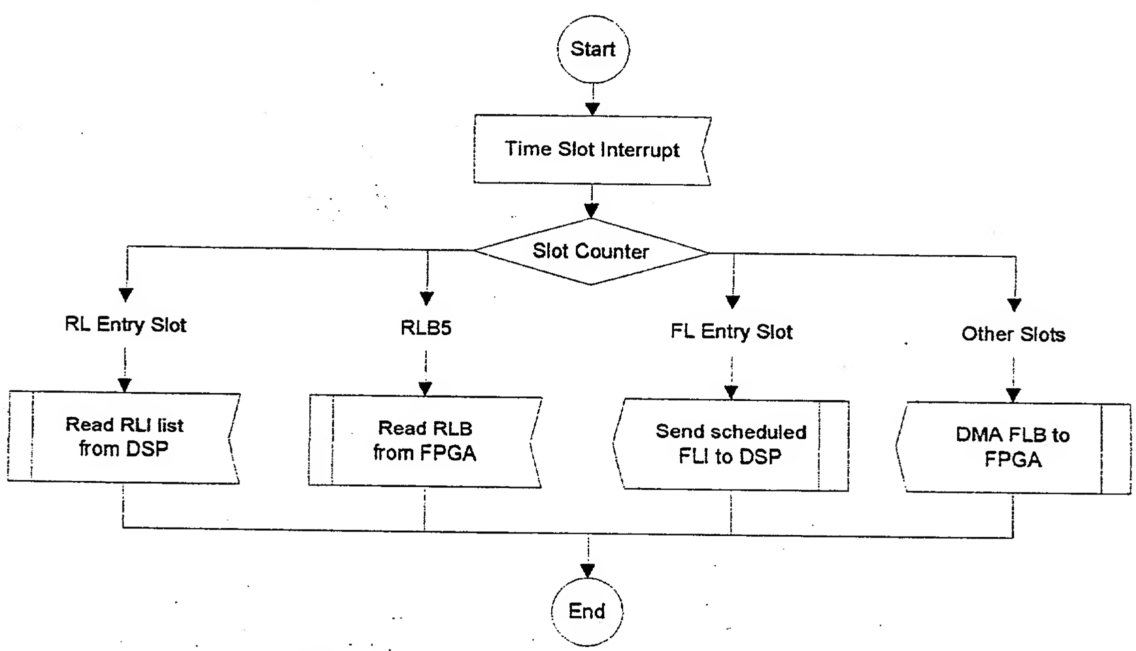
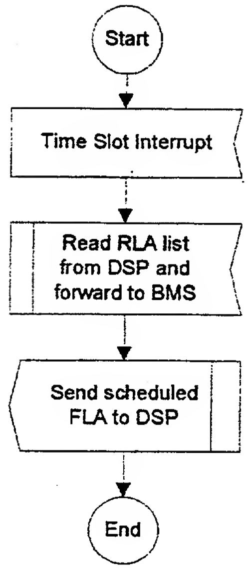


Figure ~~98~~. ~~Base partition engine – slot-driven component~~
92



93
Figure ~~99~~. Base partition engine for the access channel.

RUID	RU1	RU5	...	RU500
Backlog	1	12	...	1
Partition 0	1	1		0
Partition 1	0	1		0
...	...	...	...	...
Partition 23	1	0		0
CHANGE	-2	0		+1

Figure ~~100~~. BMS data organization
94

Condition	message	Base starting state	Base action	Prob	Base next state	channel	RU start state	RU action	Prob	RU next state
normal	FLI	closed	send FLI		starting	pass FLI	closed	detect FLI	1-Pm(FLI)	starting
	RLI	starting	detect RLI	1-Pm(RLI)	starting	pass RLI	starting	send RLI		starting
	FLT	starting	send FLT		starting	pass FLT	starting	detect FLT	1-Pm(FLT)	starting
	RLI	starting	detect RLI	1-Pm(RLI)	open	pass RLI	starting	send RLI		starting
	FLT	open	send FLT		open	pass FLT	starting	detect FLT	1-Pm(FLT)	open
			encr = 2 or more	0.99997				encr = 2 or more	0.99458	

Pm_FLI	3.00E-06
Pf_FLI	1.00E-06
Pm_FLT	2.00E-03
Pf_FLT	3.00E-03
Pm_RLI	5.40E-03
Pf_RLI	1.40E-03

Figure ~~58~~ 95: Normal Starting of a Flow.

Condition	message	Base starting state	Base action	Prob	Base next state	channel	RU start state	RU action	Prob	RU next state
RU false	FLI	closed	skip FLI		closed	empty	closed	detect FLI	Pf(FLI)	starting
detects FLI	RLI	closed	miss RLI	1	closed	pass RLI	starting	send RLI		starting
and misses	FLT	closed	skip FLT		closed	empty	starting	skips FLT	1-Pf(FLT)	starting
both FLTs	RLI	closed	miss RLI	1	closed	pass RLI	starting	send RLI		starting
	FLT	closed	skip FLT		closed	empty	starting	skips FLT	1-Pf(FLT)	closed
			encr = 0	1.00000				encr = 2	9.9E-07	

Figure 59: Error Methods During Starting a Flow.

96

Condition	message	Base starting state	Base action	Prob	Base next state	channel	RU start state	RU action	Prob	RU next state
RU misses	FLI	closed	send FLI		starting	pass FLI	closed	detect FLI	1-Pm(FLI)	starting
both FLTs	RLI	starting	detect RLI	1-Pm(RLI)	starting	pass RLI	starting	send RLI		starting
	FLT	starting	send FLT		starting	stop FLT	starting	miss FLT	Pm(FLT)	starting
	RLI	starting	detect RLI	1-Pm(RLI)	open	pass RLI	starting	send RLI		starting
	FLT	open	send FLT		open	stop FLT	starting	miss FLT	Pm(FLT)	closed
			encr = 2	0.99997				encr = 2	4.0E-06	
base misses	FLI	closed	send FLI		starting	pass FLI	closed	detect FLI	1-Pm(FLI)	starting
both RLIs	RLI	starting	miss RLI	Pm(RLI)	starting	stop RLI	starting	send RLI		starting
and RU	FLT	starting	skip FLT		starting	empty	starting	detect FLT	Pf(FLT)	starting
false detects	RLI	starting	miss RLI	Pm(RLI)	closed	stop RLI	starting	send RLI		starting
either FLT	FLT	closed	skip FLT		closed	empty	starting	detect FLT	Pf(FLT)	open
			encr = 2	2.9E-05				encr = 2	1.6E-05	
RU misses	FLI	closed	send FLI		starting	stop FLI	closed	miss FLI	Pm(FLI)	closed
FLI	RLI	starting	miss RLI	1-Pf(RLI)	starting	empty	closed	skip RLI		closed
base misses	RLI	starting	miss RLI	1-Pf(RLI)	closed	empty	closed	skip RLI		closed
both RLIs			encr = 2	3.0E-06				encr = 0	3.0E-06	
RU misses	FLI	closed	send FLI		starting	stop FLI	closed	miss FLI	Pm(FLI)	closed
FLI, and base	RLI	starting	detect RLI	Pf(RLI)	starting	empty	closed	skip RLI		closed
false detects	RLI	starting	detect RLI	Pf(RLI)	open	empty	closed	skip RLI		closed
either RLI			encr = 2	6.4E-09				encr = 0	6.0E-06	
RU false	FLI	closed	skip FLI		closed	empty	closed	detect FLI	Pf(FLI)	starting
detects FLI	RLI	closed	miss RLI	1	closed	pass RLI	starting	send RLI		starting
and	FLT	closed	skip FLT		closed	empty	starting	detect FLT	Pf(FLT)	starting
false detects	RLI	closed	miss RLI	1	closed	pass RLI	starting	send RLI		starting
either FLT	FLT	closed	skip FLT		closed	empty	starting	detect FLT	Pf(FLT)	open
			encr = 0	1.00000				encr = 2	9.0E-12	

Figure 59. Error-Methods During Starting-a-Flow Continued.